LA FANTASTICA

COMMODORE 64

PROGRAMAS PARA

GANAR MUCHO DINERO
TRACG
RECUPERAR PROGRAMAS
DESPROTEGER CARTRIDGES
CONSTRUIR RESETS

LA FANTASTICA

COMMODORE 64

EDITORIAL COMODORI
BUENOS AIRES

INTRODUCCION

Sin lugar a duda alguna tiene Ud. en sus manos el libro mas completo de programas y rutinas escrito para la Commodore 64 en idioma castellano. Las numerosas utilidades escritas en lenguaje de maquina a traves del BASIC muestran en todo su esplendor a esta poderosa computadora.

Las aplicaciones presentadas van desde las ayudas para programar y tipear programas hasta usos realmente EEFECTIVOS que justifican aun mas poseer en el hogar o en la oficina la C-64.

A fin de brindarle la maxima seguridad al tipear los programas y de pasar por el menor numero de inconvenientes hemos preparado instrucciones que resumen los errores mas frecuentes y a la vez mas catastroficos, como por ejemplo perder el programa tipeado por la impaciencia de hacerlo correr sin grabarlo previamente.

Como los programas presentados corresponden a los puntos claves de la programacion, Ud. no solo aprendera y se ubicara mejor en el mundo de la computacion sino que tambien estara orgulloso de poseer LA FANTASTICA C-64.

Autores

FEDERICO ARIAS

LUIS G. ARIAS

I.S.B.N. nro. 950-9625-00-0. Edición de 1.000 ejemplares. Se terminó de imprimir el 18/6/85 en los ta llerss gráficos Dyp. Uruguay 383. Buenos Aires.

CONSEJOS UTILES PARA TIPEAR LOS PROGRAMAS

- Utilice las facilidades que brindan los programas que le permiten agilizar y cometer menos errores, como los comandos inmediatos y el mini-teclado numerico.

- Cuando el programa supera una pagina NO lo tipee todo en forma consecutiva. Vaya grabandolo, por ejemplo cada 10 minutos. De este modo prevenimos cualquier perdida accidental o por corte de energia elecrica.

- Una vez que termina de pasar todo el programa NO lo corra con RUN. Primero grábelo para prevenir un posible bloqueo debido a errores de tipeo o el auto New de muchos programas basic-assembler.

- Los errores mas comunes en el traspaso de listados se produce por confundir un simbolo por otro. Ejemplos de esto es colocar una letra O por un cero 0, una letra I por el numero 1, la letra S por el signo $, la letra B por el numero 8, etc.

RECUPERANDO PROGRAMAS

Muchas veces un programa se borra por error o concientemente y luego nos arrepentimos y queremos recuperarlo. Para ello sirve este programa.

Ingrese el programa, corralo con RUN; luego tipee SYS(850) y el programa perdido se podra listar y grabar normalmente.

```
100 REM RECUPERANDO PROGRAMAS
110 FOR A=833 TO 938
120 READ B:POKE A,B:C=C+B:NEXT
130 IF C<>11380 THEN PRINT"EXISTE UN
ERROR EN LAS LINEAS DE DATA":STOP
140 END
200 DATA 85,78,78,69,87,32,83,89
210 DATA 83,40,56,53,48,41,32,32
220 DATA 32,160,3,200,177,43,208,251
230 DATA 200,152,24,101,43,160,255,200
240 DATA 145,43,133,45,200,165,44,145
250 DATA 43,133,46,160,255,200,177,45
260 DATA 72,240,11,200,177,45,133,46
270 DATA 104,133,45,24,144,237,200,177
280 DATA 45,208,243,104,24,165,45,105
290 DATA 2,133,45,144,2,230,46,133
300 DATA 47,133,49,165,46,133,48,133
310 DATA 50,169,0,133,51,133,52,165
320 DATA 55,133,53,165,56,133,54,76
330 DATA 116,196
```

COMANDOS INMEDIATOS

Este programa permite usar cualquiera de las letras en combinacion con las teclas SHIFT o Commodore para poner instantaneamente un comando BASIC en pantalla. Esto permite un ahorro considerable de tiempo y se cometen menos errores de tipeo.

LISTA DE COMANDOS

Letra	SHIFT	Commodore
A	PRINT	PRINT#
B	AND	OR
C	CHR$	ASC
D	READ	DATA
E	GET	END
F	FOR	NEXT
G	GOSUB	RETURN
H	TO	STEP
I	INPUT	INPUT#
J	GOTO	ON
K	DIM	RESTORE
L	LOAD	SAVE
M	MID$	LEN
N	INT	RND
O	OPEN	CLOSE
P	POKR	PEEK
Q	TAB(	SPC(
R	RIGHT$	LEFT$
S	STR$	VAL
T	IF	THEN
U	TAN	SQR
V	VERIFY	CMD
W	DEF	FN
X	LIST	FRE
Y	SIN	COS
Z	RUN	SYS

Si deseamos escribir PRINT en pantalla hacemos SHIFT A. Para escribir DATA presionamos la tecla del logotipo de Commodore y la letra D.
El programa se activa o desactiva con el SYS dado en pantalla.

```
140 IF PEEK(PEEK(56)*256)<>120THEN
POKE56,PEEK(56)-1:CLR
150 HI=PEEK(56):BASE=HI*256
160 PRINT"[clr] PACIENCIA..."
170 FOR AD=0 TO 211:READ BY
180 POKE BASE+AD,BY:NEXT AD
210 POKE BASE+26,HI: POKE BASE+81,HI
220 POKE BASE+123,HI: POKE BASE+133,HI
231 IF PEEK(65532)=34 GOTO 240
232 POKE BASE+9,72: POKE BASE+48,194
233 POKE BASE+52,235: POKE BASE+92,160
234 POKE BASE+154,72: POKE BASE+157,224
235 POKE BASE+158,234
240 PRINT"[clr] * COMANDOS VELOCES *"
250 PRINT"SI/NO :    SYSR";BASE
260 END
270 DATA 120,173,143,2,201,32
280 DATA 208,12,169,220,141,143
290 DATA 2,169,235,141,144,2
300 DATA88,96,169,32,141,143
310 DATA2,169,0,141,144,2
320 DATA88,96,165,212,208,117
330 DATA173,141,2,201,3,176
340 DATA110,201,0,240,106,169
350 DATA159,133,245,169,236,133
360 DATA246,165,215,201,193,144
370 DATA95,201,219,176,91,56
380 DATA233,193,174,141,2,224
390 DATA2,208,3,24,105,26
400 DATA170,189,159,0,162,0
410 DATA134,198,170,160,158,132
420 DATA34,160,192,132,35,160
```

```
430 DATA0,10,240,16,202,16
440 DATA12,230,34,208,2,230
450 DATA35,177,34,16,246,48
460 DATA241,200,177,34,48,17
470 DATA8,142,211,0,230,198
480 DATA166,198,157,119,2,174
490 DATA211,0,40,208,234,230
500 DATA198,166,198,41,127,157
510 DATA119,2,230,198,169,20
520 DATA141,119,2,76,220,235
530 DATA76,67,236
570 DATA153,175,199,135,161,129
580 DATA141,164,133,137,134,147
590 DATA202,181,159,151,163,201
600 DATA196,139,192,149,150,155
610 DATA191,138
650 DATA152,176,198,131,128,130
660 DATA142,196,132,145,140,148
670 DATA195,187,160,194,166,200
680 DATA197,167,186,157,165,184
690 DATA190,158,0
730 FORAD=0TO158:READ BY
740 SUMA=SUMA+BY:NEXTAD
750 IFSUMA<>20347THENPRINT"ERROR!"
```

MINI TECLADO NUMERICO

Las computadoras profesionales e incluso la Commodore 128 poseen aparte del teclado normal ("QWERTY"), un mini teclado numerico que facilita el ingreso de cifras. La utilidad de ese miniteclado es inmensa tanto en programacion como en los programas comerciales. Una vez corrido con RUN es posible adoptar, segun la necesidad momentanea, el teclado normal o el redefinido presionando CTRL N para pasar de uno a otro.

Las teclas que se redefinen y su nuevo significado son: M (0), J (1), K (2), L (3), U (4), I (5), O (6). Los numeros 7, 8 y 9 como el resto del teclado no varian.

Tambien es posible activar y desactivar el miniteclado desde un programa con POKE 50216,255 y POKE 50216,0 respectivamente.

```
10 PRINT"[clr]AGUARDE APROXIMADAMENTE 1
MINUTO"
20 FOR I=50176 TO 50261:READ X:POKE I,X
30 NEXT
40 FOR I=40960 TO 49151:POKE I,PEEK(I):
NEXT
50 FOR I=57344 TO 65535:POKE I,PEEK(I):
NEXT
60 PRINT"PRESIONANDO [CTRL N] SE PASA DE
TECLADO NORMAL A TECLADO REDEFINIDO"
70 PRINT"O VICEVERSA"
80 POKE58823,76:POKE58824,0:POKE58825,
196
90 POKE58586,169:POKE58587,1:POKE58588,
234
100 FOR I=44029 TO 44034:POKE I,234:NEXT
110 POKE1,53
500 DATA 201,14,240,65,44,40,196,240,28
,201,85,240,40,201
```

```
510 DATA 73,240,40,201,79,240,40,201,74
,240,16,201,75,240
520 DATA 16,201,76,240,16,201,77,240,28
,88,24,96,0,169
530 DATA 49,208,248,169,50,208,244,169
,51,208,240,169,52,208
540 DATA 236,169,53,208,232,169,54,208
,228,169,48,208,224,169
550 DATA 255,77,40,196,141,40,196,88
,165,198,240,252,120,76
560 DATA 180,229
```

CORRIGIENDO PROGRAMAS

Los procesadores de palabra tienen generalmente la cualidad de encontrar una palabra o caracter con lo que facilitan la correcion del texto.
Esta rutina permite encontrar la linea o lineas del programa en que aparezca la palabra buscada.
Un ejemplo comun de su uso es cuando al correr un programa vemos en pantalla un error: "Ingrese su nombe". Para encontrar la linea en donde se encuentra ese error tipografico procedemos asi:

1) Cargamos el utilitario y lo corremos con RUN
2) Ingresamos el programa a corregir.
3) Antes de la primer linea del programa ponemos
   1 "nombre"
   SYS 49152

Aparece entonces en pantalla el numero de la linea (o de las lineas) en que se halla el alfanumerico que pusimos entre comillas en la linea 1. Esta linea 1 debe ser borrada antes de guardar el programa corregido.

```
10 FORI=49152TO49255:READJ:K=K+J:
POKEI,J:NEXT
20 IFK<>16302THENPRINT"ERROR EN DATA":
STOP
30 PRINT"[clr]SYS49152 PARA BUSCAR"
100 DATA169,1,133,251,169,8,133,252,160
,0,177,251,56,229,251,56
110 DATA233,5,141,104,192,233,2,141,105
,192,160,0,177,251,170,200
120 DATA177,251,240,67,133,252,134,251,
160,0,177,251,56,229,251,170
130 DATA202,134,2,198,2,165,2,205,104,
192,48,222,133,253,173,105
140 DATA192,133,254,164,253,177,251,164
,254,217,5,8,208,229,198,253
150 DATA198,254,208,239,160,2,177,251,
170,200,177,251,32,205,189,169
160 DATA32,32,210,255,76,26,192,96
```

DOCUMENTANDO PROGRAMAS

Existen dos tipos de variables: simples y vectores. A su vez pueden ser numericas de punto flotante, enteras o alfanumericas. Es importante para la comprension de un programa documentar el uso de cada variable. El programador original puede entonces retomar con facilidad el hilo de la programaciòn y el que lo estudia puede entender el flujo del programa.

Ingrese este programa y corralo con RUN. Haga lo mismo con el programa a documentar. Quiebrelo con RUN/STOP y luego debe dar el SYS 40704. Las variables se listaran horizontalmente en pantalla.

```
100 REM ********************************
110 REM *   DOCUMENTANDO PROGRAMAS     *
120 REM ********************************
130 PA=0
140 POKE56,PEEK(56)-1
150 FOR I=40704 TO40944:READ A:POKE
I,A:PA=PA+A:NEXT
160 IF PA<>33670 THEN PRINT "EXISTE UN
ERROR EN DATA":END
170 PRINT "SYS 40704 PARA COMENZAR":NEW
180 DATA 165,45,197,47,240,93,133
190 DATA 253,165,46,133,254,160,0
200 DATA 169,0,141,61,3,177,253
210 DATA 41,128,208,60,177,253,41
220 DATA 127,32,210,255,200,173,61
230 DATA 3,201,0,208,6,177,253
240 DATA 41,128,208,46,177,253,41
250 DATA 127,32,210,255,173,61,3
260 DATA 32,210,255,169,32,32,210
270 DATA 255,152,24,105,6,144,5
280 DATA 164,254,200,132,254,168,101
290 DATA 253,197,47,240,17,208,186
300 DATA 96,169,37,141,61,3,208
```

```
310 DATA 189,169,36,141,61,3,208
320 DATA 203,165,49,197,47,240,114
330 DATA 165,47,133,253,165,48,133
340 DATA 254,160,0,169,0,141,61
350 DATA 3,177,253,240,216,41,128
360 DATA 208,77,177,253,41,127,32
370 DATA 210,255,200,173,61,3,201
380 DATA 0,208,6,177,253,41,128
390 DATA 208,63,177,253,41,127,32
400 DATA 210,255,173,61,3,32,210
410 DATA 255,169,40,32,210,255,169
420 DATA 41,32,210,255,169,32,32
430 DATA 210,255,200,177,253,24,101
440 DATA 253,197,49,240,39,177,253
450 DATA 24,101,253,170,200,177,253
460 DATA 101,254,133,254,134,253,208
470 DATA 165,96,169,37,141,61,3
480 DATA 208,172,169,36,141,61,3
490 DATA 208,186,165,48,197,50,208
500 DATA 136,96,200,234,177,253,101
510 DATA 254,197,50,240,224,16,222
520 DATA 136,208,202
```

SPRITES EN BASIC

Trabajar con sprites en el basic de la Commodore 64 es en extremo arduo. Es necesario hacer uso de POKE repetidamente a los registros de la posición del sprite. Para ahorrarse este trabajo se puede usar la rutina que utilizamos a continuacion:

direccion	Uso
50432	posibilita mover sprites
50433	sprite 0; mov. horizontal
50434	sprite 0; mov. vertical
50435	sprite 1; mov. horizontal
50436	sprite 1; mov. vertical
"	"
"	"
50447	sprite 7; mov. horizontal
50448	sprite 7; mov. vertical
50688	modifica el movimiento

```
100 REM **********************
110 REM * SPRITES EN BASIC *
120 REM **********************
130 FOR L=50880 TO 51116:READ A:POKE
L,A:NEXT
140 DATA 169,255,45,0,198,240,16,169,0,
141,0,198,162,21,189,0
150 DATA 197,157,0,198,202,208,247,162,
1,169,1,141,80,197,173,80
160 DATA 197,45,0,197,240,3,76,243,198,
232,232,14,80,197,208,238
170 DATA 76,49,234
180 DATA 169,0,29,0,197,208,3,76,97,199
,169,128,61
190 DATA 0,197,240,48,254,0,198,208,40,
222,255,207,76,144,199,80
200 DATA 197,45,16,208,208,12,173,16,
208,13,80,197,141,16,208,76
210 DATA 43,199,173,16,208,77,80,197,
```

```
141,16,208,189,0,197,157,0
220 DATA 198,76,97,199,222,0,198,208,40
,254,255,207,208,29,173,80
230 DATA 197,45,16,208,208,12,173,16,
208,13,80,197,141,16,208,76
240 DATA 91,199,173,16,208,77,80,197,
141,16,208,189,0,197,157,0
250 DATA
198,169,0,232,29,0,197,208,3,76,140,199,1
69,128,61,0
260 DATA 197,240,11,254,0,198,208,20,
222,255,207,76,134,199,222,0
270 DATA 198,208,9,254,255,207,189,0,
197,157,0,198,202,76,233,198
280 DATA 169,255,221,255,207,240,3,76,
43,199,173,80,197,76,17,199
290 DATA 120,169,192,141,20,3,169,198,
141,21,3,88,96
310 S=50433:PRINT"[clr]":H=247:V=1
320 FOR L=S TO S+16 STEP2
330 POKE L,H:POKE L+1,V
340 H=H+1:V=V+1:NEXT
350 POKE 50688,FF
360 FOR L=2040 TO 2047
370 POKE L,14:NEXT
380 FOR L=896 TO 959
390 POKE L,255:NEXT
400 S=1
410 FOR L=53287 TO 53294
420 POKE L,S:S=S+1:NEXT
430 POKE 53281,0
440 S=5:A=10
450 FOR L=53248 TO 53263 STEP 2
460 POKE L,S+48:POKE L+1,A+40
470 S=S+25:A=A+25:NEXT
480 POKE 53269,255
490 POKE 50432,255
500 SYS 51104
510 GOTO 510
```

.En los PRINT e INPUT Ud. hallarà expresiones encerradas entre corchetes. Significa que debe efectuarse lo que indican y NO copiarse literalmente.

Ejemplos: [ctrl 1] Presione las teclas CTRL y 1
[10 espacios] Presione 10 veces la barra espaciadora.

Tenga en cuenta la siguiente lista:

Expresion	Ud. Tipea	Ud. Ve
[clr]	shift clr/home	♥
[home]	clr/home	S
[arriba]	shift |crsr|	●
[abajo]	|crsr|	Q
[izq.]	shift -crsr-	|
[der.]	-crsr-	]
[ctrl 9]	ctrl 9	R
[ctrl 0]	ctrl 0	—
[ctrl 1]	ctrl 1	┐
[ctrl 2]	ctrl 2	E
[ctrl 3]	ctrl 3	£
[ctrl 4]	ctrl 4	◥
[ctrl 5]	ctrl 5	▒
[ctrl 6]	ctrl 6	↑
[ctrl 7]	ctrl 7	←
[ctrl 8]	ctrl 8	π

TRACE

El BASIC residente en memoria ROM adolece de la falta de ciertos comandos útiles como son NUMBER y TRACE. El comando TRACE es muy importante para detectar errores en un programa ya que nos permite seguir la ejecución del programa linea a linea.
Cuando ejecutemos un programa con TRACE activado, el número de cada linea ejecutada sera mostrado en pantalla.Esto le permitira a Ud. observar el camino seguido por su programa.
Esta informacion puede ayudarlo a ahorrar un gran esfuerzo en la ubicacion de errores logicos, causados por el incorrecto flujo del programa. En BASIC esos problemas pueden ser causados por equivocados números de linea en un GOTO o en un GOSUB. Tambien puede ser una incorrecta variable o en un test condicional IF.

Con TRACE Ud. puede determinar el punto inicial en el que el programa comienza a comportarse incorrectamente.
Este programa en BASIC cargará una rutina en lenguaje de maquina que nos permitira hacer uso del TRACE cuando la necesitemos sin afectar el programa que estudiamos.

Cuando el programa sea corrido, la rutina TRACE es pokeada en alta memoria, de manera que luego al cargar el programa que queremos examinar la rutina no sea borrada. En consecuencia cargue su programa en forma normal.

Utilizando TRACE:
Una vez cargado TRACE, sera activado mediante un comando SYS el cual activará la rutina. Tambien podremos desactivarlo mediante un SYS. Recordamos que SYS es siempre seguido de una dirección que

indica donde el programa ML esta alojado.
Con SYS tambien desactivaremos TRACE.
El trazado producido por la rutina consiste de una serie de números de lineas del programa, separado por espacios. Cualquier impresión producida por el programa quedarà intercalada con los números de linea.

Una característica poco usual de este TRACE es que mostrarà el resultado de cada sentencia IF ejecutada. Este resultado sera indicado mediante la impresion de una T (True=verdadero) y una F (False=falso).
Frecuentemente es muy importante saber si la parte condicional de una sentencia fue ejecutada; esta característica le brinda a Ud. un camino fàcil para verificar que el programa está realizando las decisiones correctamente.

En cualquier momento Ud. puede presionar la tecla STOP cuando el programa este corriendo, luego reasumirà la ejecucion con el comando CONT. Esto le permitirà a Ud. detener el scrolling para efectuar verificaciones. Otra posibilidad es colocar en puntos claves del programa STOPs con igual fin. Tambien es util activar el TRACE solo en las partes donde se sospecha que puedan existir errores.
La funcion TRACE no le resolverà todos los problemas derivados de errores lògicos o de programacion, pero simplifica el estudio de un listado impreso o de pantalla.

Cargando el TRACE:

Antes de correr el programa no olvide guardarlo previamente. Para activar el TRACE utilice SYS 40712 y para desactivarlo SYS 40736.

```
 1 REM ***********************************
 2 REM *        RUTINA TRACE              *
 3 REM ***********************************
 10 LM=PEEK(55)+PEEK(56)*256-248
 15 HI=INT(LM/256):LO=LM-HI*256
 20 POKE55,LO:POKE56,HI:CLR
 25 TRACE=PEEK(55)+PEEK(56)*256
 30 A=TRACE
 40 PRINT:PRINT"CARGANDO LA RUTINA TRACE
 EN" A
 50 FORD=1TO201:READN:CS=CS+N
 60 IFN>=0THENPOKEA,N:GOTO80
 70 N=TRACE+ABS(N):HI=INT(N/256):LO=N-
 HI*256:POKEA,LO:A=A+1:POKEA,HI
 80 A=A+1:NEXT
 90 HI=INT((TRACE+37)/256):LO=TRACE+37-
 HI*256:POKETRACE+5,LO:POKETRACE+9,HI
 100 IFCS<>11307THENPRINT"ERROR EN
 DATA":STOP
 110 PRINT:PRINT"TRACE LISTO.":PRINT
 120 PRINT" SYS"TRACE"= TRACE ACTIVADO"
 130 PRINT" SYS"TRACE+24"= TRACE
 DESACTIVADO"
 140 END
 500 DATA169,76,133,124,169,8,133,125,
 169,0,133
 501 DATA126,169,0,141,-245,141,-246,141
 502 DATA-247,96,169,201,133,124,169,58,
 133
 503 DATA125,169,176,133,126,96,201,32,
 208
 504 DATA3,76,115,0,72,173,-247,240,31,
 238
 505 DATA-247,201,167,240,24,169,0,141,
 -247
 506 DATA32,-211,208,4,169,84,208,2,169,
 70
 507 DATA32,210,255,169,32,32,210,255,
 32,-211
```

```
508 DATA240,109,165,57,141,-243,141,
-245,165
509 DATA58,141,-244,141,-246,201,250,
176
510 DATA89,169,0,141,-239,141,-240,
141,-241
511 DATA141,-242,142,-238,162,15,14,
-243
512 DATA46,-244,120,248,173,-239,109,
-239
513 DATA141,-239,173,-240,109,-240,
141,-240
514 DATA173,-241,109,-241,141,-241,
216,88,202
515 DATA16,216,162,2,189,-239,72,74,74
516 DATA74,74,32,-224,104,41,15,32,
-224,202
517 DATA16,236,169,32,32,210,255,174,
-238
518 DATA104,201,167,208,3,141,-247,
201,58
519 DATA176,3,76,132,0,96,165,57,205,
-245
520 DATA208,5,165,58,205,-246,96,205,
-242,208
521 DATA1,96,9,48,141,-242,76,210,255
```

ORDENAMIENTO INSTANTANEO

Una de las rutinas que no pueden faltar en la coleccion de un programador es aquella que permita la alfabetizacion o el ordenamiento numérico de una cantidad importante de elementos alfanuméricos. Su uso es practicamente obligado en los programas de aplicaciones, tanto de uso particular como comercial, ya que el ordenamiento brinda gran flexibilidad y claridad al manejo del programa.

La siguiente rutina en lenguaje de maquina, accesible desde un programa en basic, efectúa un ordenamiento de 1000 elementos en forma casi instantanea (solo 2 segundos!). Tener en cuenta la linea 300, ya que es la que pasa la información.

```
10 I=49152:SUM=0
20 READ A:IF A=256 THEN 40
30 SUM=SUM+A:POKE I,A:I=I+1:GOTO 20
40 IF SUM<>45295 THEN PRINT"ERROR EN
DATAS":END
100 PRINT"[clr]"
110 N=1000
120 DIM AA$(N)
130 PRINT "CREANDO "N" STRINGS AL AZAR"
140 SD=-TI:A=RND(SD)
150 FOR I=1 TO N
160 PRINT I"[arriba]"
170 N1=INT(RND(1)*10+1)
180 A$=""
190 FOR J=1 TO N1
200 B$=CHR$(INT(RND(1)*26+65))
210 A$=A$+B$
220 NEXT J
230 AA$(I)=A$
240 NEXT I
250 PRINT "TOQUE UNA TECLA PARA ORDENAR"
```

```
260 GET A$:IF A$="" THEN 260
270 PRINT "ORDENANDO..."
280 T1=TI
300 SYS 49152,N,AA$(K)
310 T2=TI
320 PRINT "LISTO"
330 PRINT "TOQUE UNA TECLA PARA MOSTRAR
LA LISTA ORDENADA"
340 GET A$:IF A$="" THEN 340
350 FOR I=1 TO N:PRINT I,AA$(I):NEXT
360 PRINT:PRINT N" ELEMENTOS ORDENADOS
EN"(T2-T1)/60"SEGUNDOS"
49152 DATA 32,253,174,32,158,173
49158 DATA 32,247,183,165,20,133
49164 DATA 253,165,21,133,254,32
49170 DATA 253,174,32,158,173,162
49176 DATA 1,165,71,157,85,193
49182 DATA 157,125,193,165,72,157
49188 DATA 105,193,157,145,193,165
49194 DATA 253,208,2,198,254,198
49200 DATA 253,160,3,24,189,125
49206 DATA 193,101,253,157,125,193
49212 DATA 189,145,193,101,254,157
49218 DATA 145,193,136,208,236,189
49224 DATA 85,193,133,80,189,105
49230 DATA 193,133,81,189,125,193
49236 DATA 133,82,189,145,193,133
49242 DATA 83,32,21,193,144,4
49248 DATA 202,208,228,96,165,82
49254 DATA 133,78,165,83,133,79
49260 DATA 160,2,177,78,153,250
49266 DATA 0,136,16,248,48,11
49272 DATA 24,165,80,105,3,133
49278 DATA 80,144,2,230,81,160
49284 DATA 2,177,80,153,247,0
49290 DATA 136,16,248,32,32,193
49296 DATA 144,230,56,165,82,233
49302 DATA 3,133,82,176,2,198
49308 DATA 83,32,21,193,176,31
```

```
49314 DATA 160,2,177,82,153,247
49320 DATA 0,136,16,248,32,32
49326 DATA 193,176,225,160,2,177
49332 DATA 80,145,82,185,247,0
49338 DATA 145,80,136,16,244,48
49344 DATA 183,160,2,177,80,145
49350 DATA 78,185,250,0,145,80
49356 DATA 136,16,244,24,189,85
49362 DATA 193,125,125,193,133,82
49368 DATA 189,105,193,125,145,193
49374 DATA 133,83,102,83,102,82
49380 DATA 32,21,193,176,22,189
49386 DATA 85,193,157,86,193,189
49392 DATA 105,193,157,106,193,32
49398 DATA 53,193,232,32,69,193
49404 DATA 76,71,192,189,125,193
49410 DATA 157,126,193,189,145,193
49416 DATA 157,146,193,32,69,193
49422 DATA 232,32,53,193,76,71
49428 DATA 192,165,81,197,83,208
49434 DATA 4,165,80,197,82,96
49440 DATA 160,255,200,196,247,176
49446 DATA 11,196,250,176,6,177
49452 DATA 248,209,251,240,241,96
49458 DATA 196,250,96,24,165,80
49464 DATA 105,3,157,85,193,165
49470 DATA 81,105,0,157,105,193
49476 DATA 96,56,165,80,233,3
49482 DATA 157,125,193,165,81,233
49488 DATA 0,157,145,193,96,256
```

TECLAS FUNCION

Muchos se preguntan como hacer uso de estas teclas. Es importante saber que responden a valores ASCII. Las funciones F1, F3, F5 y F7 tienen los valores 133 a 136 y las funciones F2, F4, F6 y F8 los valores 137 a 140. El siguiente programa ilustra convenientemente su aplicacion.

```
100 DATA MISION APOLO QUE LLEGO A LA
LUNA
110 DATA 10,11,12,13
120 DATA 2
130 DATA CANTIDAD DE TECLAS DE LA C64
140 DATA 55,58,66,72
150 DATA 3
160 DATA SANTA ROSA ES LA CAPITAL DE
170 DATA SAN LUIS,SAN JUAN,FORMOSA,LA
PAMPA
180 DATA 4
190 DATA "*"
200 READ Q$:IFQ$="*"THEN END
210 PRINT Q$;"--"
220 READ A$:PRINT "F1 - ";A$
230 READ A$:PRINT "F3 - ";A$
240 READ A$:PRINT "F5 - ";A$
250 READ A$:PRINT "F7 - ";A$
260 PRINT "SU RESPUESTA? ";
270 READ A
280 GET X$:IF X$=""GOTO 280
290 X=ASC(X$)
300 IF X<133 OR X>136 GOTO 280
310 X=X-132:PRINT "F";X*2-1
320 IF X=A THEN PRINT "CORRECTO!":GOTO
340
330 PRINT "MAL!"
340 GOTO 200
```

ARCHIVOS EN CASSETTE

Almacenar datos en archivos es primordial cuando se desea avanzar en aplicaciones serias. La ventaja que presentan los archivos es que se pueden actualizar desde teclado sin modificar el programa.

Los programas siguientes ejemplifican la escritura de un archivo y su lectura.

Los numeros que siguen a una sentencia OPEN significan respectivamente: el nro. de canal (puede ser cualquier nro. de 1 a 127), el nro. de aparato al que se dirige la información (Datasette:1, Video:3, Impresora:4, Drive de discos:8), la dirección secundaria (Escribir:1, Leer:0).

Es aconsejable para facilitar la lectura de un archivo colocar al principio el nro. de registros o bien "marcar" el fin del archivo (Como en los ejemplos en que se usa "-1").

Si se desea usar una unidad de discos hay que cambiar el segundo numero en los OPEN de 1 a 8.

```
100 REM   * ARCHIVOS ESCRITOS *
110 REM   * DESDE EL  TECLADO *
120 INPUT"[clr][5 abajo]NRO. DE ITEMS EN
EL ARCHIVO";N:DIM W$(N+1)
130 INPUT"[abajo]NOMBRE DEL
ARCHIVO";NOM$:IF NOM$="+1"THEN END
140 PRINT"INGRESE CADA ITEM":OPEN
1,1,1,NOM$
150 FOR X=0 TO
N:INPUTW$(X):PRINT#1,W$(X):IF
W$(X)<>"-1" THEN NEXT
160 CLOSE1:FOR X=0 TO N
PRINTW$(X):NEXT:PRINT"TOQUE UNA TECLA"
170 GETA$:IFA$=""THEN 170
180 GOTO 100
190 END
```

Antes de correr el programa lector de archivos recuerde rebobinar el cassette hasta el punto de comienzo.

```
300 REM   * LECTOR DE ARCHIVOS *
310 INPUT"[clr][8 abajo]NOMBRE DEL
ARCHIVO A LEER";NOM$
320 IF NOM$="+1"THEN END
330 OPEN1,1,0,NOM$:N=0:DIMW$(16)
340 INPUT#1,W$(N):N=N+1
350 IFW$(N-1)="-1"THEN 410
360 GOTO 340
370 CLOSE1:FOR X=0
TO(N-1):PRINTW$(X):NEXT
380 PRINT"TOQUE UNA TECLA"
390 GETA$:IFA$=""THEN 390
400 RUN 300
410 END
```

PROTECCION DE PROGRAMAS EN CASSETTE

Cuando un Programa de cassette es ingresado con LOAD o grabado con SAVE la seccion de memoria usada es la que va de la direccion 828 a 1019 (ver Mapa de Memoria en capitulo 7).
El nombre del Programa se guarda a partir de la direccion 833 y el resto se rellena con espacios.
Cuando la C64 encuentra un Programa durante un LOAD, la pantalla muestra solo los primeros 16 caracteres del nombre. Esta propiedad puede usarse para prevenir reproducciones.

(1) Grabando con caracteres ocultos: al grabar con un nombre de mas de 16 caracteres, si chequeamos uno o mas de los caracteres que no se muestran en pantalla podemos bloquear el programa si no correspomden al nombre correcto.
Ejemplo

a) grabamos el programa con 17 caracteres.
SAVE"NOMBRE[10 espacios]A"

b) incluimos la siguiente subrutina a la que vamos al principio del programa con GOSUB 10000:

```
        10000 A=PEEK(833+16):IF CHR$(A)="A"
THEN RETURN
        10100 SYS 833
```

Cuando alguien intente hacer una copia usarà NOMBRE u otro y entonces la subrutina de la linea 10000 bloquearà el programa.

SINTETIZADOR 64
Este excelente programa simula un completo panel de control de un sintetizador.
El teclado se transforma en un doble piano, que se muestra en pantalla junto con el estado de los elementos de sonido que se estan creando
Ademas de los dos teclados, se muestra en pantalla una fila indicadora (arriba) y una descripcion de la actividad de las teclas funcion:
F1 y F3: permiten subir y bajar el volumen respectivamente. Rangos: max 15, min 0
F4: cambia el estado del modo Mantenido ("M" en la fila indicadora). En este modo los sonidos no desaparecen sino se presiona otras teclas o silencio con la barra espaciadora
F6: cambia el estado del modo Multivoz ("V" en la fila indicadora). En este modo pueden ejecutarse simultaneamente hasta 3 voces.
F7 y F5: cambia estados de Caida y Acordes.
F2: permite definir las formas de ondas.
Cuando un modo esta activado, la letra que le corresponde se muestra al reverso.

OCTAVAS: se cambian presionado la tecla CTRL y un nro del 1 al 8 (octava mas baja y mas alta respectivamente).

ACORDES: puesto el modo Acordes con F5, una segunda fila indicativa aparece en la parte superior de la pantalla. A la izquierda va el nombre, que se cambia presionando SHIFT y un nro del 1 al 9: (1)Mayor, (2)Menor, (3)Disminuido, (4)Aumentado, (5)Septimo Mayor, (6)Septimo Menor, (7)Septimo Dominante, (8)Sexto Mayor, (9)Sexto Menor.
A la derecha se muestra la posicion: raiz, 1ra inversion, 2da inversion.

FORMAS DE ONDA: Para redefinir formas de onda presionar F2. La computadora pregunta por valores de Attack, Decay, Sustain, y Release y muestra a la derecha los valores actuales. Si no se desean cammbios presionar RETURN.
La siguiente pregunta es por el tipo de forma de onda, debiendo ingresar la primer letra del tipo elegido. Si selecciona Pulse, entonces se le pedirà una proporción.

Refrescando conceptos:
====================
ATTACK: es el tiempo que tarda el sonido en alcanzar el volumen mas alto.
DECAY: es el tiempo que tarda en caer hasta el sostenido (Sustain).
SUSTAIN: tiempo en que se mantiene el volumen. Un organo tiene un alto valor de Sustain.
RELEASE: determina el tiempo que le toma al sonido llegar al silencio. Un valor grande de Release es como una cuerda de guitarra que gradualmente cesa de vibrar.

Formas de Onda:
1. SAWTOOTH: diente de sierra
2. TRIANGLE: triangular
3. PULSE: cuadrada
4. NOISE: ruido

```
100 REM ******************************
110 REM *      SINTETIZADOR 64       *
120 REM ******************************
200 GOTO470
240 IFRA<0THENRA=R
250 RB=R:T=S+V*Z7:POKEVN,V*Z7:POKE
NL,FL(RA):POKENH,FH(RA):SYSS2:POKE
T+Z4,WV+Z1
260 FORI=RATORBSTEPSGN(RB-RA)/2:POKE
T,FL(I):POKET+1,FH(I):NEXT
270 IFPEEK(IK)=JANDPEEK(IK)-64THEN270
280 RA=RB:POKET+Z4,WV+P:V=V+MN*(Z1+Z3*
(V=Z2)):RETURN
310 POKEBF,Z0:FORI=Z0TOZ2:A=R+C(C1,C2,
I):POKEVN,I*Z7:POKENL,FL(A)
320 POKENH,FH(A):SYSS2:NEXT:POKES+Z4,
WV+Z1:POKES+11,WV+Z1:POKES+18,WV+Z1
330 IFPEEK(IK)=JANDPEEK(IK)-64THEN330
340 POKES+Z4,WV+P:POKES+11,WV+P:POKES+
18,WV+P:RETURN
370 A=PEEK(IK):SYSS1:J=PEEK(ET):IFJ=Z0
ORA=ZSTHENRETURN
380 FORI=Z1TOJ:R=K(PEEK(ET+I))+OC:IF
R=OCTHENNEXT:RETURN
390 T(I)=V*Z7:POKEVN,T(I):POKENL,FL(R):
POKENH,FH(R):SYSS2
400 IFMNTHENV=V+Z1:IFV=Z3THENV=Z0
410 NEXT:FORI=Z1TOJ:POKES+T(I)+Z4,WV+
Z1:NEXT
420 SYSS1:IFJ=PEEK(ET)ANDA=PEEK(IK)
THEN420
430 FORI=Z1TOJ:POKES+T(I)+Z4,WV+P:NEXT:
GOTO370
470 PRINT"[clr]"CHR$(142);CHR$(8);:
POKE53280,0:POKE53281,0:POKE788,52
480 FORI=1TO39:SP$=SP$+" ":LN$=LN$+"#":
NEXT
490 PRINT"[ctrl 8]OCTAVA=5     VOZ=1
[ctrl 2]C:S:M:[ctrl 9]V[ctrl 0]:[ctrl
```

```
9]P[ctrl 0] [ctrl 8]VOLUMEN=10[der.]"LN$
500 POKE214,23:PRINT:PRINTTAB(13)
"SINTETIZADOR  64[home][abajo][abajo]"
510 A$="POR FAVOR ESPERE[ctrl 2]":
POKE214,21:PRINT:PRINTTAB(13)"[ctrl
6]"A$: S=54272:GOSUB1580
520 DIMFL(134),FH(134),K(255),C(8,2,2):
OC=48:VL=10:MN=1:LL=1:RA=-1
530 Z1=1:Z2=2:Z3=3:Z4=4:Z7=7:ZS=64:
FF=255:HB=256
540 IK=197:BF=198:VN=251:NL=900:NH=901:
ET=829:S1=49152:S2=49408:FORI=Z1TO41
550 K(ASC(MID$("Q2W3ER5T6Y7UI9O0P-*
CHR$(92)CHR$(94)[home]ZSXDCVGBHNJM,L.:/"
, I)))=I:NEXT
560 PRINTTAB(13)"[ctrl 4][arriba]"A$:
R=5.8:A=10787.4138:J=Z2↑(-Z1/12)
570 FORI=94TO0STEP-1:FH(I)=INT(A*R/HB):
FL(I)=A*R-HB*FH(I):A=A*J:NEXT
580 PRINTTAB(13)"[arriba]"A$:GOSUB1310
610 FORI=Z0TO8:FORJ=Z0TOZ2:READC(I,J,
0),C(I,J,1),C(I,J,2):NEXT:READC$(I):NEXT
620 READNM$(0),NM$(1),NM$(2):FORI=1TO8:
READAD(I),SR(I),WV(I),PL(I),PH(I):NEXT
630 FOR R=1 TO 2:READI,J:FORA=ITOJ:READ
IN:POKEA,IN:NEXT:NEXT
640 PRINTTAB(9)"[abajo](ctrl X PARA
FINALIZAR)":I=1:GOSUB860
680 WAITBF,FF:J=PEEK(IK):GETA$:R=K
(ASC(A$))+OC:IFR=OCTHENGOSUB800:GOTO680
690 IFSLTHENGOSUB240:GOTO680
700 IFCHTHENGOSUB310:GOTO680
710 IFLLTHENGOSUB370:GOTO680
720 T=S+V*Z7:POKEVN,V*Z7:POKENL,FL(R):
POKENH,FH(R):SYSS2:POKET+Z4,WV+Z1
730 IFMNTHENV=V+Z1:IFV=Z3THENV=Z0
740 IFPEEK(IK)=JANDPEEK(IK)-64THEN740
750 POKET+Z4,WV+P:WAITBF,FF:GETA$:J=
PEEK(IK):R=K(ASC(A$))+OC:IFR-OCTHEN720
```

```
760 GOSUB800:GOTO680
800 IFCH=0THEN830
810 FORI=0TO2:IFA$=MID$("CHR$(91)CHR$
(93))",I+1,1)THENC2=I:PRINT"[home]
[abajo]"TAB(22)NM$(I):RETURN
820 NEXT:A=ASC(A$):IFA>32ANDA<42THEN
C1=A-33:PRINT"[home][abajo]"TAB(12)
C$(C1):RETURN
830 FORI=1TO8:IFA$<>MID$("[ctrl 1][ctrl
2][ctrl 3][ctrl 4][ctrl 5][ctrl 6][ctrl
7][ctrl 8]",I,1)THENNEXT:GOTO850
840 OC=12*(I-Z1):PRINT"[home]"TAB(7)
MID$(STR$(I),2):RETURN
850 FORI=1TO8::IFA$<>MID$("[C 1][C 2] [C
3][C 4][C 5][C 6][C 7][C 8]",I,1)THEN
NEXT:GOTO880
860 POKE902,PL(I):POKE903,PH(I):WV=
WV(I):POKE904,WV:POKE905,AD(I):POKE
906,SR(I)
870 PRINT"[home]"TAB(16)MID$(STR$(I),
2):RETURN
880 IFA$<>"[f1]"ANDA$<>"[f3]"THEN930
890 VL=VL-(VL<15ANDA$="[f1]")+(VL>0
ANDA$="[f3]"):POKES+24,VL
900 PRINT"[home]"TAB(37)RIGHT$("0"+
MID$(STR$(VL),2),2):RETURN
930 IFA$="[f4]"THENP=1-P:POKE1047,13+
128*P:GOTO1580
940 IFA$="[f6]"THENMN=1-MN:POKE1049,22+
128*MN:GOTO1580
950 IFA$="[f8]"THENLL=1-LL:POKE1051,16+
128*LL:RETURN
960 IFA$="[f7]"THENSL=1-SL:RA=-1:POKE
1045,19+128*SL:CH=1:GOTO990
970 IFA$<>"[f5]"THEN1010
980 POKE1045,19:SL=0
990 CH=1-CH:POKE1043,3+128*CH:IFCH=0
THENPRINT"[home][abajo]"LN$:PRINTSP$:
RETURN
```

```
1000 PRINT"[home][abajo]"SP$"[der.]
[arriba]TIPO
ACORDE:"C$(C1)TAB(23)NM$(C2)
"INVERSION[der.]"LN$:RETURN
1010 IFA$=" "THENGOSUB1580:RA=-1:POKE
BF,Z0:RETURN
1020 IFA$=""THENGOSUB1580:PRINT"[clr]";
:POKE788,49:END
1030 IFA$<>"[f2]"THENRETURN
1070 GOSUB1470:POKE214,13:PRINT
1080 PRINT"VOZ A DEFINIRSE (1-8)";:J=1:
GOSUB1500
1090 IFIN<1ORIN>8THENGOSUB1470:GOTO 1400
1100 I=IN:PRINTTAB(31)"ATT:"MID$(STR$
(INT(AD(I)/16)),2)
1110 PRINTTAB(31)"DEC:"MID$(STR$(AD(I)
AND15),2)
1120 PRINTTAB(31)"SUS:"MID$(STR$(INT(
SR(I)/16)),2)
1130 PRINTTAB(31)"REL:"MID$(STR$(SR(I)
AND15),2)
1140 PRINTTAB(31)"WVF:[ctrl 4]"MID$(
"SAWTRIPULNSE",3*LOG(WV(I))/LOG(2)-11,3)
"[ctrl 2]
1150 IFWV(I)=64THENPRINTTAB(31)"PLS:
"MID$(STR$(PH(I)*HB+PL(I)),2)
1180 POKE214,14:PRINT:PRINT"PROP.
ATTACK(0-15)";:J=2:GOSUB1500:IFER
THEN1070
1190 AD=IN:PRINT"PROP. RATE (0-15)";:
GOSUB1500:IFERTHEN1070OSUB1500:IFER
THEN1070
1200 AD=AD*16ORIN:PRINT"NIVEL SUSTAIN
(0-15)";:GOSUB1500:IFERTHEN1070HEN1070
1210 SR=IN:PRINT"PROP. RATE (0-15)";:
GOSUB1500:IFERTHEN10700:IFERTHEN1070
1220 SR=SR*16ORIN:PRINT"[ctrl 4]S[ctrl
2]AW [ctrl 4] T[ctrl 2]RIANGLE[ctrl 4]
P[ctrl 2]ULSE[ctrl 4] N[ctrl 2]OISE[ctrl
```

```
2]";:J=1:GOSUB1500
1230 FORJ=1TO4:IFIN$<>MID$("STPN",J,1)
THENNEXT:GOTO1070
1240 WF=2↑(J+3):IFWF<>64THEN1260
1250 PRINT"PROP. RATE (0-4095)";:J=4:
GOSUB1500:PU=IN:IFIN<0ORIN>4095THEN1070
1260 WV(I)=WF:PL(I)=PU-HB*INT(PU/HB):
PH(I)=INT(PU/HB):AD(I)=AD:SR(I)=SR
1270 GOSUB1470:GOSUB1410:GOTO860
1310 POKES+24,VL:PRINT"[home][3 abajo]"
TAB(9)" [ctrl 9] [der.] [der.] ] [der.]
[der.] [der.] ] [der.] [der.] ] [der.]
[der.] [der.] "
1320 PRINT"TECLADO   [ctrl 9] [ctrl
0]2[ctrl 9] [ctrl 0]3[ctrl 9] ] [ctrl
0]5[ctrl 9] [ctrl 0]6[ctrl 9] [ctrl
0]7[ctrl 9] ] [ctrl 0]9[ctrl 9] [ctrl
0]0[ctrl 9] ] [ctrl 0]-[ctrl 9] [ctrl
0]£[ctrl 9] S "
1330 PRINT" BAJO       [ctrl 9] ] ] ] ] ]
] ] ] ] ] ] ] ] ] ] "
1340 PRINTTAB(10)"[ctrl 9]Q]W]E]R]T]Y]U
]I]O]P]]*]↑] "
1350 PRINTTAB(11)"[abajo] [ctrl 9]
[der.] [der.] ] [der.] [der.] [der.] ]
[der.] [der.] [ctrl 0] "
1360 PRINT"TECLADO      [ctrl 9] [ctrl
0]S[ctrl 9] [ctrl 0]D[ctrl 9] ] [ctrl
0]G[ctrl 9] [ctrl 0]H[ctrl 9] [ctrl
0]J[ctrl 9] ] [ctrl 0]L[ctrl 9] [ctrl
0]:[ctrl 9] [ctrl 0] "
1370 PRINT" AGUDO        [ctrl 9] ] ] ] ]
] ] ] ] ] [ctrl 0] "
1380 PRINTTAB(12)"[ctrl 9]Z]X]C]V]B]N]M
],].]/[ctrl 0] "
1410 POKE214,13:PRINT:PRINT"F1 -- [ctrl
4]MAS ALTO[ctrl 2]  F2 -- [ctrl 4]FORMA
DE ONDA[ctrl 2]
1420 PRINT"[abajo]F3 -- [ctrl 4]MAS
```

```
BAJO[ctrl 2]  F4 -- [ctrl
4]MANTENIDO[ctrl 2]
1430 PRINT"[abajo]F5 -- [ctrl
4]ACORDES[ctrl 2]  F6 -- [ctrl 4]MULTI
VOZ[ctrl 2]
1440 PRINT"[abajo]F7 -- [ctrl 4]CAIDA
[ctrl 2]    F8 -- [ctrl 4]POLIFONICO
[ctrl 2]":RETURN
1470 POKE214,12:PRINT:FORJ=1TO11:PRINT
SP$:NEXT:RETURN
1500 IN$="":PRINT"?";
1510 PRINT"[ctrl 9] [ctrl 0][izq.]";:
WAITBF,FF:GETA$:IFA$="X"THEN1020
1520 A=ASC(A$):IFA=13THENPRINT" ":
IN=VAL(IN$):ER=(IN<0ORIN>15)ORIN$="":
RETURN
1530 IFA=20ANDLEN(IN$)THENPRINT" [2 h";
:IN$=LEFT$(IN$,LEN(IN$)-1)
1540 IF(AAND127)<35ORLEN(IN$)=JTHEN1510
1550 PRINTA$;:IN$=IN$+A$:GOTO1510
1580 FORI=4TO18STEP7:POKES+I,0:NEXT:
FORI=0TO23:POKES+I,0:NEXT:RETURN
1620 DATA,4,7,,3,8,,5,9,"MAYOR      ",
,3,7,,4,9,,5,8,"MENOR      "
1630 DATA,3,6,,3,9,,6,9,"DISMINUIDO",,4
,8,,4,8,,4,8,"AUMENTADO "
1640 DATA,4,11,,4,11,,4,11,"MAYOR 7MO "
,,3,10,,3,10,,3,10,"MENOR 7MO "
1650 DATA,4,10,,4,10,,4,10,"DOMIN 7MO",
4,7,9,4,7,9,4,7,9,"MAYOR 6TA"
1660 DATA3,7,9,3,7,9,3,7,9,"MENOR 6TA",
" RAIZ  ","PRIMER",SEGUNDO
1690 DATA,249,16,,,,249,32,,,,249,64,
160,15,,249,128,,,,240,16,,,204,204,16,,
1700 DATA,252,64,200,,192,240,32,,
1730 DATA49152,49294,120,169,,141,61,3,
170,169,254,133,252,165,252,141,
,220,173
1740 DATA1,220,157,143,192,232,56,
```

```
38,252,176,239,162,,160,,189,143,192,
42,176
1750 DATA29,72,132,253,138,10,10,10,5,
253,168,185,79,192,238,61,3,172,61,3,153
1760 DATA61,3,104,192,3,240,12,164,253,
200,192,8,208,219,232,224,8,208,209,88
1770 DATA96,17,135,134,133,136,29,13,
20,0,69,83,90,52,65,87,51,88,84,70,67,54
1780 DATA68,82,53,86,85,72,66,56,71,
89,55,78,79,75,77,48,74,73,57,44,64,
58,46
1790 DATA45,76,80,43,47,94,61,1,19,59,
42,92,3,81,2,32,50,4,95,49
1820 DATA49408,49454,169,212,133,252,
169,,160,6,145,251,136,145,251,170,
169,8
1830 DATA136,145,251,138,145,251,136,
192,1,208,249,188,41,193,185,132,3,
145,251
1840 DATA232,224,6,208,243,96,2,3,,1,6,5
```

SECRETARIA ELECTRONICA

Cuando tenemos una computadora con la cual aprender y divertirnos, tambien le buscamos una aplicaciòn en el hogar o la oficina. Este programa no solo nos permite mantener archivos de inventario y agenda telefonica, sino que se comporta realmente como una secretaria al proceder a DISCAR el numero de la persona elegida.
Esta posibilidad surge del fantastico circuito de sonido de la Commodore 64, que produce los tonos exactos requeridos
Cuando el programa se corre con RUN, la primera pantalla permite elegir entre 2 modos: Agenda telefonica o Inventario. En cualquiera de los modos aparece un menu de 9 opciones para el manejo de los datos en pantalla y archivo.
Cada opcion elegida es facil de seguir y se permite guardar los datos en cassette o diskette.
Cada archivo abierto puede manejar hasta 60 registros de 5 campos cada uno.
En el numero de opcion 5 se usan los datos de modo diferente. Si se esta en modo Agenda Telefonica, la computadora generara los tonos necesarios para efectuar la llamada telefonica (telfonos digitales). Se puede tambien rediscar y cronometrar el tiempo de la llamada.
En el modo Inventario esa misma opcion 5 brinda un total de los valores.
Es posible asimismo efectuar una salida por impresora de los registros con la opcion 8.

```
100 REM **************************
110 REM * SECRETARIA ELECTRONICA *
120 REM **************************
170 DIMA$(5,60),O$(5),C$(6),P1(3),P2(3)
,B$(2,5)
180 S=54272:SZ=0
190 PO=1
200 READ P1(1),P1(2),P1(3),P2(1),P2(2),
P2(3)
210 FORI=1TO5:READB$(1,I):NEXT
220 FORI=1TO5:READB$(2,I):NEXT
230 DATA 697,770,852,1210,1340,1481
240 DATA "NOMBRE    :","TELEFONO :",
"DIRECCION:","CIUDAD    :","PROVINCIA :"
250 DATA "ITEM :","VALOR:","UBICACION
:","FECHA COMPRA:","NRO. SERIE   :"
260 RF=2000:RF=(RF-30)/8:RH=INT(RF/8):
RL=RF-8*RH
270 FORX=STOS+24:POKEX,0:NEXT
280 POKES+23,3:POKES+21,RH:POKES+22,RL
290 PRINT"[ctrl 1][clr][2 abajo]":
POKE53281,1
300 PRINTTAB(6)"**************************
*****"
310 PRINTTAB(6)"** SECRETARIA ELECTRONI
CA **"
320 PRINTTAB(6)"**************************
*****[15 abajo]"
340 PRINTTAB(8)"1. AGENDA TELEFONICA":
PRINTTAB(8)"[abajo]2. INVENTARIO"
350 GOSUB2530
360 IFK=49 THENOP=1:GOTO390
370 IFK=50 THENOP=2:GOTO390
380 PRINT"CORREGIR 1 O 2":GOTO350
390 N=0:GOTO550
410 PRINT"[2 abajo]ENTRE":PRINT"[abajo]
[der.]F PARA FINALIZAR":PRINT"[der.]C
PARA RE-INGRESAR":G1=1:G2=2
420 I=N+1
```

```
430 IF I<60 OR I=60 THEN 460
440 PRINT"[2 abajo] *** ARCHIVO
COMPLETO (N=60)***[2 abajo]"
450 GOSUB2520:RETURN
460 PRINT" "
470 PRINTB$(OP,1);:A$(1,I)="":INPUT
A$(1,I)
480 IF A$(1,I)="F"ORA$(1,I)="FIN"
THEN1720
490 IF A$(1,I)=""THEN470
500 IF A$(1,I)<>"R"THEN520
510 I=I-1:N=I-1:PRINT" *** RE-INGRESE
ULTIMO BLOQUE ***":GOTO460
520 FORZ=2 TO 5
530 PRINTB$(OP,Z);:A$(Z,I)="":INPUT
A$(Z,I):NEXTZ
540 N=I:GOTO410
550 SC=4:PRINT"[clr][2 abajo][6 der.]**
SECRETARIA ELECTRONICA **"
560 IFOP=2 THEN580
570 PRINTTAB(11)"AGENDA TELEFONICA":
GOTO590
580 PRINTTAB(15)"INVENTARIO"
590 PRINT"**********************************
***********"
600 PRINT"APRIETE":PRINT"1. AGREGAR
DATOS"
610 PRINT"2. CAMBIAR DATOS":PRINT"3.
MOSTRAR LISTA DE NOMBRES"
620 PRINT"4. MOSTRAR DE UNO":PRINT"5.
USAR DATOS"
630 PRINT"6. GUARDAR ARCHIVO":PRINT"7.
TRAER ARCHIVO"
640 PRINT"8. LISTAR EN IMPRESORA":
PRINT"9. FIN"
650 GOSUB2530
660 IF K<49 OR K>57 THEN 550
670 PRINT"[clr]"
```

```
680 ON (K-48)GOSUB 700,730,900,970,1890
,1520,1260,2290,2340
690 GOTO550
700 GOSUB410
710 RETURN
730 INPUT "NRO. DE REGISTRO";M$:IFM$=""
THENM$="1"
740 PRINT"[2 abajo]USE":PRINT"[abajo]
CURSOR PARA CAMBIOS":PRINT"[abajo] /E/
PARA ELIMINAR EL ITEM"
750 PRINT"[abajo] < RETURN > SI NO
REALIZA CAMBIOS":PRINT"[2
abajo]":GOSUB1160
760 G1=1:I=M
770 PRINTA$(1,M);:P$="":INPUT P$
780 IF P$="" THEN 830
790 IF P$<>"/E/" THEN 820
800 FORZ=1 TO 5:A$(Z,M)="←ULTIMO
REGISTRO DEL ARCHIVO":NEXT
810 GOSUB1720:N=N-1:RETURN
820 A$(1,M)=P$:G2=1
830 FOR Z=2 TO5:PRINTA$(Z,M);:P$="":
INPUT P$
840 IF P$="" THEN 860
850 A$(Z,M)=P$:G2=G2+1
860 NEXTZ
870 GOSUB 1040
880 ON T GOTO770,730,890
890 RETURN
900 IF G2<>0 THENGOSUB1720
910 FORXX=1 TON:M=35-LEN(A$(2,XX)):
T$=". "
920 PRINTTAB(4-LEN(T$))XX;"[izq]";T$;
A$(1,XX);TAB(M);A$(2,XX)
930 I1=INT(XX/10):IFI1<>XX/10THEN 950
940 GOSUB2520
950 NEXTXX:GOSUB2520:RETURN
970 M$="":INPUT"[2 abajo]NRO. DE
REGISTRO";M$:IFM$=""THEN1030
```

```
980 GOSUB1160
990 PRINT"[clr][3 abajo][3 der.]"
1000 FORI=1TO5:PRINT"[5 der.]";A$(I,M):
NEXT
1010 GOSUB1040
1020 ONT GOTO990,970,1030
1030 RETURN
1040 PRINT"[2 abajo]APRIETE[abajo]":
PRINT"A PARA UN REGISTRO ANTERIOR"
1050 PRINT"[abajo]P PARA PROXIMO
REGISTRO":PRINT"[abajo]O PARA TRAER OTRO
REGISTRO":GOSUB2520
1060 T=3:IFK<>65 THEN1090
1070 T=1:IFM=1 THEN1140
1080 M=M-1
1090 IF K<>80 THEN1120
1100 T=1:IFM=NTHEN 1140
1110 M=M+1
1120 IF K<>79 THEN 1140
1130 T=2
1140 RETURN
1160 IF ABS(ASC(M$)-53)>4 THEN 1200
1170 M=VAL(M$):IF M<N OR M=N THEN 1190
1180 M=N
1190 RETURN
1200 FORI=1 TO N:M=I
1210 IF LEN(M$)>LEN(A$(1,I))THEN 1230
1220 IF M$=LEFT$(A$(1,I),LEN(M$))THEN
1240
1230 NEXTI
1240 RETURN
1260 PRINT"ENTRE 1. CASSETTE"
1270 PRINT"      2. DISKETTE"
1280 INPUT "[2 abajo]OPCION";AS
1290 IFAS<1 OR AS>2 THEN1260
1300 IF AS=2 THENAS=2:SS=8:GOTO1320
1310 AS=0:SS=1
1320 H$=CHR$(13):INPUT"NOMBRE DEL
ARCHIVO":N$
```

```
1330 IF AS<>2 THEN 1370
1340 REM
1350 OPEN2,SS,AS,"0:"+N$+",S,R"
1360 GOTO1380
1370 OPEN2,SS,AS,N$
1380 INPUT#2,O,NT,F$,D$
1390 IF O=OP THEN 1460
1400 IF O=2 THEN 1420
1410 PRINT"[abajo]ARCHIVO TELEFONICO":
GOTO1430
1420 PRINT"[abajo]ARCHIVO DE INVENTARIO"
1430 PRINT"PROCEDE (S/N)?"
1440 GOSUB2530:IF K=78 THEN RETURN
1450 IF K<>83 THEN 1440
1460 OP=O:N=NT
1470 PRINT"ANTERIOR ACTUALIZACION:";D$
1480 FORI=1 TON:FORJ=1 TO5
1490 INPUT#2,A$(J,I):NEXT J:NEXTI
1500 CLOSE2:GOSUB2520:RETURN
1520 IF G2<>0 THEN GOSUB1720
1530 PRINT"ENTRE 1. CASSETTE"
1540 PRINT"      2. DISKETTE"
1550 INPUT "[2 abajo]SU ELECCION";AS
1560 IF AS<1 OR AS>2 THEN 1530
1570 IF AS=2 THEN AS=2:SS=8:GOTO1590
1580 AS=1:SS=1
1590 H$=CHR$(13):INPUT"NOMBRE DEL
ARCHIVO";N$
1600 INPUT "FECHA.";D$
1610 IF AS<>2 THEN 1650
1620 REM
1630 OPEN2,SS,AS,"0:"+N$+",S,W"
1640 GOTO 1660
1650 OPEN 2,SS,AS,N$
1660 PRINT#2,OP;H$;N;H$;S$;H$;D$;H$
1670 FOR I=1 TO N:FORX=1 TO 5:IF
A$(X,I)="" THEN A$(X,I)=" "
1680 PRINT#2,A$(X,I);H$:NEXT X:NEXT I
1690 PRINT#2,;:CLOSE2
```

```
1700 G1=0 :RETURN
1720 PRINT"ORDENAMIENTO SEGUN QUE
CAMPO?"
1730 FOR Z=1 TO 5:PRINTZ;". ";B$(OP,Z):
NEXT Z
1740 INPUT F
1750 IF F<1 OR F>5 THEN 1720
1760 G2=0
1770 PRINT"[3 abajo]***** ORDENANDO
DATOS *****[3 abajo]"
1780 FOR I=1 TO N-1:FOR J=N TO I+1 STEP
-1
1790 IF A$(F,I)<A$(F,J) OR A$(F,I)=
A$(F,J) THEN 1810
1800 GOSUB1860
1810 NEXT J
1820 IF A$(F,I)<>A$(F,I+1) THEN 1840
1830 I=I+1
1840 NEXT I
1850 RETURN
1860 FOR Z=1 TO 5:O$(Z)=A$(Z,I):
A$(Z,I)=A$(Z,J)
1870 A$(Z,J)=O$(Z):NEXT Z:RETURN
1890 IF OP=1 THEN 1960
1900 SM=0
1910 FOR I=1 TON:SM=SM+VAL(A$(2,I)):
NEXTI
1920 PRINT"[2 abajo][4 der.]COSTO TOTAL
DE TODOS LOS ITEMS":PRINT TAB(14)"[2
abajo]";SM
1930 GOSUB 2520:RETURN
1960 GOSUB 970:IF M$="" THEN 2100
1970 PRINT"[clr][2 abajo][2 der]"
1980 T$=A$(2,M):L=LEN(T$)
1990 FOR I=1 TO 5:PRINT"[2 der]";
A$(I,M):NEXT:PRINT"[2 abajo][15 der.]";
2000 IF L<10 THEN 2020
2010 L=L+1:T$="1"+T$
2020 FORJ=1 TO L:P$=MID$(T$,J,1)
```

```
2030 PRINTP$;:IF ASC(P$)<48 OR ASC(P$)>
57 THEN 2090
2040 T=VAL(P$):IF T<>0 THEN 2060
2050 F2=941:F1=1336:GOSUB2410:GOTO2080
2060 I=INT((T-1)/3)+1:IJ=T-3*(I-1)
2070 F1=P1(I):F2=P2(IJ):GOSUB2410
2080 FORX=1TO 250:NEXT
2090 NEXTJ
2100 PRINT"[2 abajo]":PRINT"[der.]
APRIETE":PRINT"[abajo]"
2110 PRINT"[der.]R PARA RE-DISCAR":
PRINT" C PARA CRONOMETRAR"
2120 PRINT"[der.]N PARA UN NUEVO NUMERO"
2130 GOSUB 2520
2140 IFK=82 THEN1970
2150 IF K=78 THEN  1960
2160 IF K<>67 THEN2180
2170 GOSUB2560:GOSUB2200
2180 RETURN
2200 TI$="000000"
2210 FORI=1 TO 6:C$(I)=MID$(TI$,I,1):
NEXT
2220 PRINT"[home][22 abajo][16 der.]";
2230 PRINTC$(1);C$(2);":";C$(3);C$(4);
":";C$(5);C$(6)
2240 FORI=1 TO 250:NEXT
2250 I=PEEK(197):IF I=64 THEN 2210
2260 IF I=17 THEN1970
2270 RETURN
2290 OPEN4,4
2300 FORI=1 TON:FORJ=1 TO5
2310 PRINT#4,A$(J,I):NEXT
2320 PRINT#4," ":PRINT#4," ":NEXT
2330 PRINT#4,;:CLOSE4:RETURN
2340 PRINT"[clr][2 abajo][der.]DESEA
TERMINAR EL PROGRAMA Y PERDER"
2350 PRINT"[der.]TODA LA INFORMACION EN
MEMORIA?"
2360 PRINT"[der.][abajo] (S/N)?"
```

```
2370 GOSUB2530
2380 IF K=78 THEN710
2390 IF K<>83 THENRETURN
2400 END
2410 FD=INT(F1/.06097):H1=INT(FD/256):
L1=FD-(256*H1)
2420 FD=INT(F2/.06097):H2=INT(FD/256):
L2=FD-(256*H2)
2430 POKES+5,0:POKE S+12,0
2440 POKES+6,224:POKES+13,240
2450 POKE S+24,31
2460 POKES+1,H1:POKES,L1
2470 POKES+8,H2:POKES+7,L2
2480 POKES+4,33:POKES+11,33
2490 FORZX=1 TO 100:NEXT
2500 POKE S+4,32:POKES+11,32
2510 POKES+24,0:RETURN
2520 PRINT"[abajo][4 der.]TOQUE UNA
TECLA PARA CONTINUAR"
2530 GET K$:IF K$<>"" THEN 2530
2540 GET K$:IF K$="" THEN 2540
2550 K=ASC(K$):RETURN
2560 PRINT"[4 arriba]";:FORI=1TO 2:
PRINT"[34 espacios]":NEXT:RETURN
```

Estadistica en Ayuda del Azar

APLICACION A LA QUINIELA

El siguiente es un programa de aplicación, mediante el mismo tratamos de determinar, basandonos en los resultados obtenidos en cada sorteo, los números con mayores posibilidades de salir sorteados. Partimos de la premisa de que cuanto mayor es la demora, mayor es la probabilidad de que salga sorteado.

Por razones de importancia consideramos para el cálculo sólo los primeros cinco numeros de cada sorteo, teniendo en cuenta que influyen factores tales como la capacidad de la maquina y la velocidad de ejecución.

Debemos tener presente que los datos deben ser actualizados, para ello debemos agregar cada sorteo en forma de DATA al final del programa. El primer numero del DATA corresponde al primer premio, recordar ademas que el ultimo DATA debe ser siempre F,I,N,A,L .

El programa presenta un menù con dos opciones, la primera DEMORAS A LA CABEZA Y A LOS CINCO : calcula cuantos sorteos llevan sin aparecer los numeros del 0 al 9 en la unidad, decena y centena.
La segunda opción PROPUESTAS: realiza una interpretación de las demoras y con los nùmeros favoritos compone un nùmero propuesta con ayuda del comando RND.

Este programa utiliza como datos los sorteos de la QUINIELA NACIONAL, el ùltimo sorteo incluido corresponde al dia martes 4 de junio de 1985.

```
1 REM ***********************************
2 REM *   ESTADISTICA DE QUINIELA       *
3 REM ***********************************
4 DIM UNI(100),DEC(100),CEN(100),
U$(900),D$(900),EST$(900)
5 READ EST$(EE):IF EST$(EE)<>"F" THEN
EE=EE+1:GOTO 5
6 RESTORE
7 PRINT"[clr][ctrl 8][4 abajo]          1)
CABEZA Y UNA CIFRAS A LOS 5"
8 PRINT"[5 abajo]          2) APUESTA
PROPUESTA"
15 INPUT "[4 abajo]       INGRESE
ELECCION";A:   ON A GOTO 49,48,600,900
48 IF SE=1 AND A=2 THEN 394
49 SE=1
50 PRINT"[clr]":PRINT"       ESTADISTICA
[ctrl 4]QUINIELERA[ctrl 2]":PRINT"[3
abajo][ctrl 6]CALCULANDO...[ctrl 2]"
60 READ A$,B$,C$,D$,E$
70 SUM$=A$+B$+C$+D$+E$
80 IFSUM$="FINAL" THEN 104
82 UNI$=MID$(A$,3,1):DEC$=MID$(A$,2,1):
CEN$=MID$(A$,1,1)
85 NUN=VAL(UNI$):NDE=VAL(DEC$):NCE=
VAL(CEN$)
90 FOR N=0 TO 9 :UNI(N)=UNI(N)+1:
DEC(N)=DEC(N)+1:CEN(N)=CEN(N)+1:NEXT N
100 UNI(NUN)=0:DEC(NDE)=0:CEN(NCE)=0
103 GOTO 60
104 IF A=2 THEN 394
105 FORI=0TO9:UNI$(I)=STR$(UNI(I))
106 DEC$(I)=STR$(DEC(I)):CEN$(I)=
STR$(CEN(I)):NEXT I:GOTO 394
190 PRINT"[clr][ctrl 2] ---------------
---    [ctrl 4] -------------"
192 PRINT"[ctrl 2]--    A LA CABEZA   --
  [ctrl 4] - A LOS CINCO -"
200 PRINT"[ctrl 2]       CEN    DEC    UNI
```

```
[ctrl 4]        DEC   UNI ":PRINT
205 FOR I=0 TO 9:UC(I)=INT(UC(I)/5):
DC(I)=INT(DC(I)/5)
210 PRINT"[ctrl 8]";I;"[ctrl 2]-";
TAB(5);CEN$(I);TAB(10);"-"DEC$(I);
TAB(16);"-"UNI$(I);
215 PRINT TAB(24);"[ctrl 6]";I;"[ctrl
2]-";TAB(29);DC(I);TAB(34);"-"UC(I):NEXT
300 PRINT:PRINT: PRINT"[ctrl 6]TOQUE C
TECLA PARA CONTINUAR"
302 GET W$:IF W$<>"C"THEN 302
305 GOTO 6
394 FOR II=0 TO 9
395 FOR FF=EE-1TO 0 STEP -1:U$(FF)=
MID$(EST$(FF),3,1)
396 IF VAL(U$(FF))<>II THEN UC(II)=
UC(II)+1:NEXT FF
397 NEXTII
400 FOR II=0 TO 9
405 FOR FF=EE-1TO 0 STEP -1:D$(FF)=
MID$(EST$(FF),2,1)
410 IF VAL(D$(FF))<>II THEN DC(II)=
DC(II)+1:NEXT FF
420 NEXTII:IF A=1 THEN  GOTO 190
440 FOR I=0 TO 9:UC(I)=INT(UC(I)/5):
DC(I)=INT(DC(I)/5):NEXT
445 UMAX=UNI(0):DMAX=DEC(0):CMAX=
CEN(0):MAXUC=UC(0):MXDC=DC(0)
450 FOR I=1 TO 9
460 IF UC(I)>UC(MAXUC) THEN MAXUC=I
465 IF DC(I)>DC(MXDC) THEN MXDC=I
470 IF UNI(I)>UMAX THEN UMAX=UNI(I):U=I
475 IF DEC(I)>DMAX THEN DMAX=DEC(I):D=I
480 IF CEN(I)>CMAX THEN CMAX=CEN(I):C=I
485 NEXT
490 PRINT"[clr][abajo]          [ctrl 6]
PROPUESTAS A LA CABEZA   [ctrl 4]"
495 PRINT"[2 abajo]UNIDAD ";U;:U1$=
STR$(INT(RND(-TI)*9+1)):U2$=STR$(INT(
```

```
RND(0)*9+1))
496 PRINT"          ";U1$+U2$+STR$(U)
500 PRINT"[abajo]DECENA ";D;:D1$=STR$(
INT(RND(0)*9+1)):D2$=STR$(INT(RND(0)*
9+1))
502 PRINT"          ";D1$+STR$(D)+D2$
505 PRINT"[abajo]CENTENA";C;:C1$=STR$(
INT(RND(0)*9+1)):C2$=STR$(INT(RND(0)*
9+1))
507 PRINT"          ";STR$(C)+C1$+C2$
508 PRINT"[2 abajo]     FAVORITOS A LOS
CINCO"
510 PRINT"[abajo]UNIDAD  ";MAXUC
515 PRINT"[abajo]DECENA  ";MXDC
520 PRINT"[2 abajo][ctrl 8]   FAVORITOS
DOS CIFRA A LA CABEZA  [ctrl 4]"
525 CU$=STR$(MAXUC):CD$=STR$(MXDC)
530 PRINT"[2 abajo]             ";STR$
(D)+CU$;"         ";CD$+STR$(U)
535 PRINT"[abajo]  [ctrl 1]ENTRE M PARA
MENU T PARA TERMINAR"
540 INPUT M$:IF M$="M" THEN 6
545 IF M$<>"T" THEN 540
550 END
1800 DATA 606,591,124,350,410
1805 DATA 886,615,304,605,040
1810 DATA 506,363,549,501,863
1815 DATA 078,639,195,831,837
1820 DATA 408,782,055,780,747
1825 DATA 172,490,189,697,485
1830 DATA 962,840,382,171,813
1835 DATA 586,900,388,806,867
1840 DATA 419,581,191,273,469
1845 DATA 046,516,560,142,536
1850 DATA 581,266,264,664,160
1855 DATA 795,695,877,325,679
1860 DATA 317,424,149,232,025
1865 DATA 504,445,333,897,646
1870 DATA 005,687,188,104,496
```

```
1875 DATA 505,753,751,423,476
1880 DATA 171,113,391,906,137
1885 DATA 474,544,601,479,601
1890 DATA 487,070,042,448,561
1895 DATA 076,625,845,918,512
1900 DATA 544,466,050,063,344
1905 DATA 385,400,836,586,182
1910 DATA 678,400,836,586,182
1915 DATA 800,494,939,446,492
1920 DATA 392,935,162,130,062
1925 DATA 174,481,524,135,128
1930 DATA 286,914,476,531,333
1935 DATA 029,951,292,200,142
1940 DATA 292,903,573,516,399
1945 DATA 304,717,111,708,017
1950 DATA 053,872,869,199,294
1955 DATA 453,580,519,450,742
1960 DATA 384,618,437,266,088
1965 DATA 911,545,821,866,234
1970 DATA 134,312,965,874,163
1975 DATA 728,356,438,689,004
1980 DATA 127,695,672,397,842
1985 DATA 931,979,074,410,326
1990 DATA 576,989,425,850,111
1995 DATA 282,671,915,871,014
2000 DATA 636,219,728,936,222
2005 DATA 963,608,848,278,615
2010 DATA 900,509,042,082,268
2015 DATA 542,915,638,230,786
2020 DATA 312,185,119,794,139
2025 DATA 485,425,528,200,169
2030 DATA 605,666,640,783,351
2035 DATA 687,974,111,607,539
2040 DATA 751,248,067,448,636
2045 DATA 669,157,792,013,826
2050 DATA 359,401,150,802,069
2055 DATA 856,014,863,835,688
2060 DATA 151,250,417,619,710
2065 DATA 092,243,714,048,662
```

```
2070 DATA 323,702,279,243,708
2075 DATA 156,685,687,503,816
2080 DATA 075,809,961,491,153
2085 DATA 811,456,900,894,259
2090 DATA 713,482,332,783,205
2095 DATA 692,493,797,264,196
2100 DATA 912,551,034,706,327
2105 DATA 294,640,022,471,238
2110 DATA 506,826,877,447,482
2115 DATA 155,666,262,032,253
2120 DATA 481,948,246,802,216
2125 DATA 438,853,675,274,530
2130 DATA 043,289,466,216,495
2135 DATA 059,770,660,934,179
2140 DATA 035,511,140,406,430
2145 DATA 072,314,691,425,798
2150 DATA 098,735,922,525,887
2155 DATA 502,655,630,162,656
2160 DATA 468,594,415,230,515
2165 DATA 370,845,247,820,656
2170 DATA 605,070,874,376,454
2175 DATA 377,318,837,562,785
2180 DATA 928,470,846,758,007
2185 DATA 977,653,436,815,684
2190 DATA 299,359,793,255,160
2200 DATA 997,789,902,049,318
2205 DATA 105,688,137,608,540
2210 DATA 794,414,591,547,568
2215 DATA 975,957,101,601,818
2220 DATA 412,659,349,101,895
2225 DATA 580,974,863,743,771
2230 DATA 889,663,197,186,137
2235 DATA 256,808,666,716,424
2240 DATA 745,492,678,561,436
2245 DATA 772,858,947,807,627
2250 DATA 618,247,334,578,431
2255 DATA 204,145,971,806,242
2260 DATA 786,537,534,302,399
2265 DATA 082,420,288,007,172
```

```
2270 DATA 822,159,134,393,387
2275 DATA 072,850,706,795,981
2280 DATA 454,631,169,989,580
2285 DATA 589,177,320,369,452
2290 DATA 017,109,551,780,569
2295 DATA 926,631,526,974,337
2300 DATA 078,364,139,042,530
2305 DATA 500,569,267,668,886
2310 DATA 135,384,218,486,894
2315 DATA 878,940,669,386,380
2320 DATA 436,999,992,284,697
2325 DATA 542,709,472,351,465
2330 DATA 064,060,131,475,972
2335 DATA 180,465,634,945,447
2340 DATA 954,657,184,209,674
2345 DATA 669,066,098,298,955
2350 DATA 782,983,923,804,468
2355 DATA 282,664,367,317,253
2360 DATA 814,477,818,715,058
2365 DATA 473,146,566,019,863
2370 DATA 319,633,062,087,678
2375 DATA 243,725,744,726,424
2380 DATA 694,536,067,049,598
2385 DATA 300,451,812,402,254
2390 DATA 148,590,225,452,141
2395 DATA 547,898,742,621,043
2400 DATA 178,010,974,755,484
2405 DATA 873,283,372,240,320
2410 DATA 417,571,039,364,452
2415 DATA 264,400,191,204,258
2420 DATA 131,418,580,019,662
2425 DATA 831,396,778,141,861
2430 DATA 033,992,545,800,072
2435 DATA 283,611,601,497,045
2440 DATA 733,557,367,108,959
2445 DATA 811,651,158,854,806
2450 DATA 279,348,499,084,212
2455 DATA 795,089,995,367,395
2460 DATA 104,317,486,308,674
```

```
2465 DATA 551,102,430,961,941
2470 DATA 411,053,304,267,712
2475 DATA 659,223,361,814,325
2480 DATA 775,134,738,755,516
2485 DATA 073,993,328,553,522
2490 DATA 988,876,319,414,307
2495 DATA 073,184,768,738,191
2500 DATA 177,985,902,643,728
2505 DATA 878,814,865,673,446
2510 DATA 891,911,354,030,718
2515 DATA 873,820,145,334,604
4000 DATA F,I,N,A,L
```

Si Ud. desea tener estos programas o algunos de ellos en cassette o disket, puede solicitarlos a sus autores, al teléfono

48-3103

DESPROTECCION DE CARTRIDGES

Muchos cartridges pueden ser facilmente desprotegidos y copiados a disco o cassette.
Solo será necesario agregar a la computadora una llave doble inversora (switch) de muy bajo costo.

Antes de comenzar es importante tener en cuenta que se debe proceder cuidadosamente ya que se puede afectar el cartridge o la computadora si se lo inserta cuando esta se encuentra encendida.
Si bien la instalación del switch es sencilla, es aconsejable que lo realice una persona con ciertos conocimientos en electronica o un tecnico.
El switch sera colocado en la parte posterior derecha de la computadora, cerca del cartridge port.

Pasos iniciales:
1- Comprar una llave doble inversora (switch).
2- Desconectar todos los cables que van a la computadora. Abra la computadora sacando los tornillos en la parte inferior y localice el cartridge port.
3- Encuentre un espacio libre a la derecha del cartridge port y cuidadosamente efectue un agujerito con un taladro u otra herramienta que nos permitira instalar el switch. Tenga sumo cuidado con los cables.
4- Los pins estan numerados desde el centro de la computadora hacia la derecha. Será necesario localizar los pins 2,3 y 8 de la fila superior; no sera necesario modificar los pins de la fila inferior.
5- Los pins 2 y 3 tienen 5 Voltios. Ambos pins estan conectados al mismo lugar del circuito impreso. Ud. debera cortar ambos pins (2 y 3). Suelde la mitad inferior (lado del circuito

impreso) de los pins 2 y 3 juntos. Luego suelde la mitad superior (lado del conector) de los pins 2 y 3 juntos.
6- Instale cables en un lado del switch provenientes de la mitad superior de los pins 2 y 3 y del otro lado del switch los que vienen de la parte inferior.
7- Desconecte el pin 8 de la fila superior del conector; instale cables desde el switch con las mitades superior e inferior del pin 8.
8- Verifique que una posicion del switch conectara los pins 2 y 3; la otra conecta al pin 8. Instale el switch y cierre la computadora.
9- Conecte todos los cables y a Trabajar.

Los pins 2 y 3 contienen 5 Volts y el pin 8 se denomina GAME Line del cartridge. Estos tres pins deben estar desconectados para permitir que el cartridge sea insertado sin que este tome control de la computadora cuando sea encendida.

Pasos a seguir:
==============
(A) Con la computadora y el switch apagados inserte el cartridge y luego encienda la computadora.
(B) Ud. vera 30719 BASIC BYTES FREE, el cartridge no tomo control de la computadora pero su presencia ha sido detectada.
(C) Cargue un programa monitor Assembler (ML) en $C000.
(D) Encienda el switch.
(E) El monitor de ML debe aun controlar la computadora. Si en la pantalla sucede algo anormal, por ejemplo que aparezcan caracteres raros, entonces no sera posible copiar el cartridge en forma simple.
(F) Utilice el comando M del monitor para examinar la memoria comenzando en $8000.

(G) Si el cartridge es un "auto start" Ud. encontrara algo similar a:

```
.8000 09 80 70 80 C3 C2 CD 38
.8008 30
```

Los primeros dos bytes contienen los vectores de la direccion del cold-start del cartridge ($8009 en el ejemplo). Los siguientes dos bytes contienen los warm-start vectors del cartridge ($8070 en el ejemplo). Los siguientes tres bytes son las letra CBM. Los ultimos dos bytes deben ser 80 en hexadecimal.
Cuando se encienda la computadora esta examinara si estos nueve bytes estan ubicados en $8000. Si es asi la computadora pasa el control al cartridge.

El cold-start vector es utilizado cuando se enciende la computadora y el warm-start vector es utilizado por la tecla RESTORE.
Estos vectores indican la direccion adecuada para cada cartridge en particular y pueden No ser los del ejemplo. Lo que es importante es el CBM80. La computadora debe ver estos cinco bytes. Si el cartridge es auto-start no importa si esta presente o no (fisicamente). Si estos cinco bytes estan cuando la computadora es reseteada (ver RESET en seccion miscelaneas) o cuando la tecla RESTORE es presionada, el control pasara a los vectores. Este es el motivo por el cual los programas en cartridge que han sido pasados a disco o cassette correran cuando la computadora sea reseteada (SYS 64738). CBM80 actuara desde el RAM o ROM en forma identica.

Encendiendo la computadora con el switch apagado evitaremos que la computadora reconozca la informacion contenida en el cartridge.

La computadora sabra que hay algo en la ubicacion de la memoria del cartridge pero no podra leer la informacion e indicara 30719 BYTES FREE.
Los cartridge normalmente ocupan la memoria $8000 a $9FFF (cartridge de 8K). Guardar la informacion del cartridge a disco o cassette es muy simple.

(H) Utilice el comando S (SAVE) del monitor ML para guardar la memoria.

```
 S 8000 A000"nombre" 08  (01 cassette)
```

$8000 es la direccion inicial del cartridge, $A000 es la direccion final mas uno del bloque de memoria a ser copiado. Siempre agregar un byte de informacion, ya que en general los monitores no copian el ultimo byte.

La computadora puede ser ahora apagada y el cartridge sacado. El programa puede ser cargado desde el disco o cassette. Utilice 8,1 (disco) o 1,1 (cassette) para que la informacion sea cargada en la direccion de memoria correspondiente.
Trate presionando la tecla RESTORE; si el programa no empieza a ejecutarse utilice SYS 64738. Normalmente esto correra el programa.
Ocasionalmente algunos programas no funcionaran. Sucede que algunos cartridge tienen una proteccion en lenguaje de maquina.

Los programas de cartridge estan contenidas en chips ROM. Esto significa que la informacion contenida en el chip puede ser solo leida. Ninguna informacion puede ser escrita.
La memoria ROM (Read Only Memory) puede ser considerada como memoria permanente y no puede ser modificada de ninguna manera.
Lo que algunos cartridges intentan hacer es

modificar su propia memoria cuando el programa es ejecutado. Como el programa esta contenido en ROM la memoria no podra ser alterada por lo que el programa es ejecutado sin problemas. Pero cuando el programa ha sido copiado en disco o cassette y luego cargado en la computadora , el programa reside ahora en memoria RAM. Los programas residentes en RAM pueden ser modificados en cualquier momento. Ahora cuando el programa sea ejecutado podra alterarse a si mismo.
Para poder encontrar la proteccion del cartridge sera necesario cargar y ejecutar el monitor ML.

1) Cargue el monitor.
2) Cargue el programa desde el disco o cassette.
3) Cambie CBM80 a CBM00. Esto evitarà que el programa se ejecute accidentalmente. Recuerde cambiarlo nuevamente antes de guardar el programa.
4) Transfiera el bloque de memoria $8000-$A000 a $4000.

```
T 8000 A000 4000
```

Esto le permitirà tener un area en la memoria con la cual comparar el programa original.
5) Encuentre los vectores cold y warm-start del cartridge. Utilice el comando G para ejecutar el cartridge. Intente con los vectores cold-start primero. Puede ser necesario intentar con los cold-start vectors despues de recargar el programa.

```
G 8009          en el ejemplo precedente
```

6) Despuès que el programa se detenga o bloquee, resetear la computadora. Utilice SYS 49152 para correr nuevamente el monitor ML.

7) Utilice el comando C (COMPARE) para hallar las memorias que fueron alteradas con la ejecucion del programa. Ud. ya transfiriò el programa $4000 en el punto cuatro. En general la protecciòn modificarà solo unos pocos bytes de memoria o un bloque.

```
C 8000 A000 4000
```

8) Copie las direcciones de memoria que hayan sido modificadas. Si ha sido modificado un bloque copie el area en general. Esto nos darà un area donde comenzar a investigar.

Si solo unò o dos bytes en memoria han sido modificados, el programa probablemente realizara algo similar al siguiente ejemplo. Si la memoria alterada fuera $8400 y su valor fue cambiado a cero (00) Ud. encontraria el codigo:

```
.8040 A9 00    LDA #$00
.8042 8D 00 84 STA $8400
```

El programa en ROM no serà afectado por esto ya que no puede ser sobre escrito. Pero como estamos en RAM $8040 sera afectado.
Para encontrar el codigo Ud. puede utilizar el comando D (DISASSEMBLER) e investigar el programa linea por linea. Cada vez que observe que el programa no carga nada como en $8042 seguramente es parte del esquema de proteccion.

En nuestro ejemplo, para prevenir la modificacion de $8400 serà necesario cambiar el codigo desde $8042 hasta $8045 a NOP (NO OPERATION) comando que no hace nada. Entonces ya no serà modificado $8400 y el programa funcionarà normalmente.
Algunos programas utilizaran este metodo de alteraciòn en distintos puntos; hay que

buscarlos; no es dificil, solo lleva un poco de tiempo.

```
.8000 09 80    ORA #$80
.8002 D4       ???
.8003 80       ???
.8004 C3       ???
.8005 C2       ???
.8006 CD 38 30 CMP $3038
.8009 A9 00    LDA #$00
.800B 8D 00 80 STA $8000
.800E A9 20    LDA #$20
.8010 8D 00 84 STA $8400
.8013 A9 60    LDA #$60
.8015 8D 00 90 STA $9000
.8018 A9 00    LDA #$00
.801A 8D 00 94 STA $9400
.801D 8D 00 95 STA $9500
.8020 8D 20 95 STA $9520
.8023 8D FF 80 STA $80FF
.8026 A9 F7    LDA #$F7
.8028 A2 1F    LDX #$1F
.802A A0 00    LDY #$00
.802C 8C 0E DC STY $DC0E
.802F 8C A9 00 STY $00A9
```

El programa precedente cambia distintas RAM directamente. Este metodo se denomina direccionamiento directo (DIRECT Addressing) o direccionamiento absoluto (ABSOLUTE Addressing). Los valores en las memorias pueden ser tambien cambiados mediante 1) direccionamiento indexado (INDEXED Addressing), 2) INDIRECT INDEXED Addressing o 3) INDEXED INDIRECT Addressing.

```
.8070 A9 00    LDA #$00
.8072 A0 10    LDY #$10
.8074 99 40 81 STA $8140,Y  /  STA $8140+Y
```

```
.8077 C8        INY          /  Y=Y+1  ($11)
.8078 99 40 81  STA $8140,Y  /  STA $8140+Y
.808B A9 42     LDA #$42     /  comienzo Prog.
```

Este es un ejemplo de INDEXED addressing. Es muy facil de entender, ya que lo que hace es sumar el valor de Y al valor de la direccion de memoria obteniendo asi la nueva direccion. El valor de Y puede ser cualquier valor entre 00 y FF, por lo que la nueva dirección estará dentro de los 255 bytes a partir de $8140 en el ejemplo.
INDEXED addressing es tan facil de "arreglar" como DIRECT addressing. Cambie los bytes $8074 hasta $8076 por NOP, tambien desde $8078 hasta $807A.
La dirección puede ser indexada con X o Y.

Veamos ahora INDIRECT INDEXED addressing. Este tipo de indexacion utiliza solo el registro Y y es bastante comun. INDIRECT INDEXED addressing puede ser usado para cambiar algunos bytes o un bloque de memoria.

```
.8080 A9 00     LDA #$00     /  comienza en $8200
.8082 85 B8     STA $B8      /  bajo byte 00
.8084 A9 82     LDA #$82
.8086 85 B9     STA $B9      /  alto byte 82
.8088 A0 00     LDY #$00
.808A A9 FF     LDA #$FF
.808C 91 B8     STA ($B8),Y / STA en $8200+Y
.808E C8        INY          / Y=Y+1
.808F D0 FB     BNE $808C   / IF Y<>0 THEN $808C
.8091 A9 00     LDA #$00     / comienza Prog.
```

INDIRECT INDEXED addressing utiliza la pagina cero (zero page) para guardar la verdadera dirección como un vector. La dirección debe ser

guardada en el formato desglosado en bajo byte (low byte) y alto byte (high byte) que es el utilizado por el microprocesador 6510.

Cuando la computadora ejecuta un STA ($B8),Y guarda el contenido del acumulador en la direcciòn contenida en $B8 y $B9 mas Y. El programa ha guardado la direccion $8200 en $B9 y $B8; Y inicialmente vale 00. Por lo que la computadora primero guardarà el contenido del acumulador (STA) en $8200. Luego Y es incrementada; cuando Y sea distinta de cero el programa se dirigirà a la direcciòn $808C y realizarà la operaciòn nuevamente. Esta vez Y vale 1, por lo que la nueva direcciòn donde pondremos el contenido del acumulador (STA) serà $8201. Esto continuarà hasta que Y alcance el valor FF (255). Cuando incremente $FF en uno, tendremos nuevamemte $00; entonces la sentencia BNE encontrarà que Y es igual a cero, por lo que pasarà a la siguiente instrucciòn.
Como resultado de toda esta rutina la memoria desde $8200 a $82FF ha sido llenada con 00.

Si Ud. encuentra que su programa tiene una rutina de proteccion similar a la descripta, hay varias maneras de prevenir que el programa se auto-borre:

(a) Cambiando el alto byte en $B9 de $82 por $20 (o cualquier otro valor), fuera del area de memoria del cartridge.

(b) Sería cambiar los primeros tres bytes de la rutina por un comando JMP (JUMP) que salte la rutina de protecciòn:

```
.8080 4C 91 80 JMP $8091
```

(c) Un tercer metòdo seria cambiar los codigos de $808C y $808D por NOPs:

```
.808A A9 FF     LDA #$FF
.808C EA        NOP
.808D EA        NOP
.808E C8        INY
```

Cualquiera de estos mètodos hara el trabajo correctamente. Elija uno y vea si existe un còdigo similar en el programa en el que Ud. esta trabajando.
Recuerde que no afectarà en nada a la computadora si Ud. experimenta con el còdigo.

Un tercer mètodo de direccionamiento es el INDEXED INDIRECT. Este tipo de indexaciòn utiliza solo el registro X. Es el mètodo mas complicado.

```
.8040 A9 00     LDA #$00     /  bajo byte -$10
.8042 85 FB     STA $FB      /  STA en $00FB
.8044 A9 71     LDA #$71     /  alto byte -$10
.8046 85 FC     STA $FC      /  STA en $00FC
.8048 A2 10     LDX #$10     /  X+ bajo y alto b.
.804A A9 00     LDA #$00
.808C 81 FB     STA ($FB,X)  /  STA en $8110
.8050 A9 AF     LDA #$AF     /  comienza prog.
```

La direcciòn deseada es $8110; el bajo byte $10, el alto byte $81. El valor del registro X es restado del alto y bajo byte. Estos nuevos valores son guardados en la pàgina cero. Cuando el programa es corrido el valor de X es sumado a ambos bytes (bajo y alto).

Hay que tener presente que muchos programas pueden utilizar memoria por debajo de $8000 para

guardar cierta informaciòn (por ejemplo: High Scores) pero no con la intenciòn de protecciòn. Por lo que hay que prestar atenciòn fundamentalmente a aquel còdigo que pueda alterar la direcciòn $8000 para arriba (hasta $C000).

CARTRIDGES DE 16 K

Existen cartridges de 8K de memoria ($8000 a $9FFF) y los de 16K ($8000 a $BFFF).
Los cartridges de 16K son similares a los de 8K. La principal diferencia es que la memoria ocupada por el interprete BASIC es ahora ocupada por el cartridge.
Cuando el cartridge esta colocado y encendemos la computadora esta busca CBM80 (ya comentado). Si la computadora encuentra CBM80, entonces chequeara el contacto #8 (GAME) del cartridge-port para detectar la presencia del cartridge. Si el cartridge utiliza 16K la computadora colocara el valor $36 en la direccion $0001 para desactivar el interprete BASIC ya que el cartridge utilizara esa area de memoria.
Si Ud. sospecha que el cartridge es de 16K, coloque $36 en la direccion $0001.
Si existe codigo de maquina por encima de $A000 entonces el cartridge es de 16K.
Estos cartridges se copian a disco o cassette de la misma manera que los de 8K, salvo que el area de memoria es mayor:

```
S 8000 C000 "nombre" 08  (cass. 01)
```

Los mètodos de protecciòn son similares a los de 8K, salvo que en los cartridges de 8K habrà uno o dos esquemas de proteccion y en 16K seguramente

tres o cuatro.
Utilice las mismas tecnicas que en 8K; recuerde que ahora es necesario desactivar el BASIC, el KERNAL o ambos antes de ejecutar el programa.
Esto evitara cualquier posibilidad de interferencia entre el programa y el sistema operativo de su computadora.
A continuacion mostraremos como proceder.
El primer programa es un ejemplo de como desactivar el BASIC en lenguaje de maquina y el segundo para desactivar el KERNAL:

```
.0A00 A9 36    LDA ##$36  /  desactiva BASIC
.0A02 85 01    STA $01
.0A04 4C 09 80 JMP $8009  /  comienza el prog.
```

```
.0A00 A9 35    LDA ##$35  /  desactiva KERNAL
.0A02 85 01    STA $01
.0A04 4C 09 80 JMP $8009
```

Estos programas son completamente relocat:ables y pueden ser utilizados en cualquier memo[illegible]. El comando JMP (JUMP) se determina para cada programa.

Es importante, para concluir, que con estos mètodos y una dosis de paciencia llevarà a feliz termino cualquier desprotecciòn de cartridges.

MAPA GENERAL DE LA MEMORIA

HEX	DECIMAL	DESCRIPCION
0000	0	Registro direccion de chip
0001	1	Control y memoria de cinta
0003-0004	3-4	Float-Fixed vector
0005-0006	5-6	Fixed-Float vector
0007	7	Busca caracter
0008	8	Bandera de comillas
0009	9	Guarda columna del TAB
000A	10	0=LOAD, 1=VERIFY
000B	11	Entrar puntero del buffer
000C	12	Bandera de DIM asumda
000D	13	tipo:FF=string, 00=numerico
000E	14	tipo:80=entero, 00=p.flot.
000F	15	Data scan
0010	16	Bandera de funcion
0011	17	0=INPUT;$40=GET;$98READ
0012	18	Signo arco tangente
0013	19	Actual bandera I/O
0014-0015	20-21	Valor entero
0016	22	Puntero:lugar moment.string
0017-0018	23-24	Ultimo mom.string vector
0019-0021	25-33	Lugar para mom.strig
0022-0025	34-37	Area del puntero utilit.
0026-002A	38-42	Area para multipliccion
002B-002C	43-44	Puntero:com. del Basic
002D-002E	45-46	Puntero:com. de variables
002F-0030	47-48	Puntero:com. de vectores
0031-0032	49-50	Puntero:fin de vectores
0033-0034	51-52	Puntero:lugar de strings
0035-0036	53-54	Puntero de string util
0037-0038	55-56	Puntero:limite de memoria
0039-003A	57-58	Actual nro linea de Basic
003B-003C	59-60	Anterior nro de linnea

003D-003E	61-62	Puntero para el CONT
003F-0040	63-64	Actual nro linea deDATA
0041-0042	65-66	Actual direccion deDATA
0043-0044	67-68	Vector INPUT
0045-0046	69-70	Nombre de variable usada
0047-0048	71-72	Direccion variable usada
0049-004A	73-74	Puntero para FOR/NEXT
004B-004C	75-76	Puntero save Basic
004D	77	Simbolo comparacion
004E-0053	78-83	Area trabajo extra
0054-0056	84-86	Vector para funciones
0057-0060	87-96	Area extra numerica
0061	97	Acum 1:Exponente
0062-0065	98-101	Acum 2:Mantisa
0066	102	Acum 3:Signo
0067	103	Puntero constante eval.
0068	104	Acum 1:orden alta
0069-006E	105-110	Acum 2:Exp,Mant,Signo
006F	111	Compara signos
0070	112	Acum 1:orden baja
0071-0072	113-114	Long. cassete buffer
0073-008A	115-138	Subrutina CHRGET
007A-007B	122-123	Puntero de Basic
008B-008F	139-143	Valor RND
0090	144	Status word
0091	145	Keyswitch PIA
0092	146	Constante de tiempo(cass.)
0093	147	LOAD=0, VERIFY=1
0094	148	Salida serie
0095	149	Caracter serie
0096	150	Final de cinta
0097	151	Registro a guardar
0098	152	Nro de archivos abiertos
0099	153	Entrada de unidad
009A	154	Salida unidad CMD
009B	155	Paridad caracter cassete
009C	156	Bandera byte recibido
009D	157	RUN=0, Modo Directo$80
009E	158	Primer lectura de cassete

009F	159	Segunda lectura
00A0-00A2	160-162	Reloj Jiffy HML
00A3	163	Bit serie
00A4	164	Cuenta lazos
00A5	165	Cuenta caida
00A6	166	Puntero del buffer cassete
00A7	167	Cabeza grabación
00A8	168	Error lectura
00A9	169	Comienzo escritura
00AA	170	Scan de cassette
00AB	171	Largo escritura
00AC-00AD	172-173	Puntero:cassete buffer
00AE-00AF	174-175	Direccion fin de programa
00B0-00B1	176-177	Constantes de tiempo
00B2-00B3	178-179	Puntero:comienzo del buffer
00B4	180	1=contador habilitado
00B5	181	Próximo bit por RS232
00B6	182	Error de lectura de carac.
00B7	183	Caracteres nombre archivo
00B8	184	Corriente archivo logico
00B9	185	Corriente dirección 2ria.
00BA	186	Unidad funcionando
00BB-00BC	187-188	Puntero al archivo
00BD	189	Lectura de caracter entrada
00BE	190	Nro de bloques restantes
00BF	191	Word buffer en serie
00C0	192	Interlock motor de cassete
00C1-00C2	193-194	Dirección comienzo I/O
00C3-00C4	195-196	Puntero Kernal
00C5	197	Última tecla presionada
00C6	198	Nro de caracteres en buffer
00C7	199	Bandera de reverso
00C8	200	Fin de linea de INPUT
00C9-00CA	201-202	Fila, columna cursor
00CB	203	Tecla, nro 64 si ninguna
00CC	204	0=cursor flasheante
00CD	205	Cuenta de caida
00CE	206	Carácter bajo el cursor
00CF	207	Cursor parpadeante

00D0	208	Ingreso de Pantalla/teclado
00D1-00D2	209-210	Puntero linea Pantalla
00D3	211	Posicion del cursor
00D4	212	Bandera modo comillas
00D5	213	Longitud linea de Pantalla
00D6	214	Fila del cursor
00D7	215	Ultima entrada
00D8	216	Nro de INSERTs
00D9-00F2	217-242	Tabla enlace de lineas
00F3-00F4	243-244	Puntero color de Pantalla
00F5-00F6	245-246	Puntero de teclado
00F7-00F8	247-248	Puntero:entrada RS2l2
00F9-00FA	249-250	Puntero:salida RS232
0100-010A	256-266	Punto flotante al ASCII
0100-013E	256-318	Error de cassette
0100-01FF	256-511	Area del esPacio Procesador
0200-0258	512-600	Buffer ingreso de Basic
0259-0262	601-610	Tabla archivo logico
0263-026C	611-620	Tabla de unidad
026D-0276	621-630	Tabla direccion secundaria
0277-0280	631-640	Buffer del teclado
0281-0282	641-642	Comienzo memoria BASIC
0283-0284	643-644	ToPe memoria BASIC
0285	645	Bandera serial bus
0286	646	Actual codigo de color
0287	647	Color en el cursor
0288	648	Page de memoria de Pantalla
0289	649	Maximo esPacio de teclado
028A	650	RePite todos las teclas
028B	651	RePite contador veloz
028C	652	RePite contador retrasado
028D	653	Shift de teclado
028E	654	Ultimo Patron de shift
028F-290	655-656	Puntero:tabla de teclado
0291	657	Modo shift de teclado
0292	658	0=scroll habilitado
0293	659	Registro control RS232
0294	660	Registro comando RS232
0295-0296	661-662	Bit timing

0297	663	RS232 status register
0298	664	Nro de bits a enviar
0299-029A	665-666	Codigo de velocidad RS232
029B	667	Fin de ingreso buffer RS232
029C	668	Comienzo de ingreso buffer
029D	669	Comienzo de salida buffer
029E	670	Fin de salida buffer RS232
029F-02A0	671-672	IRQ I/O
02A1	673	CIA2 (NMI)
02A2	674	CIA1 Timer A
02A3	675	CIA1 Interrupt
02A4	676	CIA1 bandera Timer A
02A5	677	Marcador de linea pantalla
02C0-02FE	704-766	(Sprite 11)
0300-0301	768-769	Mensaje de error asociado
0302-0303	770-771	Comienzo Basic warm
0304-0305	772-773	Basic crunch
0306-0397	774-775	Mustra aviso asociado
0308-0309	776-777	Comienzo nuevo codigo Basic
030A-030B	778-779	Toma elemento aritmetico
030C	780	6510 accumulator
030D	781	6510 registro X
030E	782	6510 registro Y
030F	783	6510 status register
0310	784	USR instruccion jump
0311-0312	785-786	USR direccion jump
0313	787	sin uso
0314-0315	788-789	Hardware Interrupt vector
0316-0317	790-791	Break Interrup vector
0318-0319	792-793	NMI Interrupt vector
031A-031B	794-795	Open vector
031C-031D	796-797	Close vector
031E-031F	798-799	Input vector
0320-0321	800-801	Output vector
0322-0323	802-803	Restore I/O vector
0324-0325	804-805	INPUT vector
0326-0327	806-807	OUTPUT vector
0328-0329	808-809	Stop vector test
032A-032B	810-811	GET vector

032C-032D	812-813	Abortado I/O vector
032E-032F	814-815	Comienzo warm vector
0330-0331	816-817	LOAD vector
0332-0333	818-819	SAVE vector
0334-033B	820-827	sin uso
033C-03FB	828-1018	CASSETTE BUFFER
0340-037E	832-894	(Sprite 13)
0380-03BE	896-958	(Sprite 14)
03C0-03FE	960-1022	(Sprite 15)
0400-07FF	1024-2047	Memoria de pantalla
0800-9FFF	2048-40959	Memoria Ram BASIC
8000-9FFF	32768-40959	ROM cartridge
A000-BFFF	40960-49151	ROM Basic
A000-BFFF	49060-49151	RAM
C000-CFFF	49152-53247	Memoria RAM
D000-D02E	53248-53294	Video Chip (6566)
D400-D41C	54272-54300	Sound Chip (6581 SID)
D800-DBFF	55296-56319	Memoria color nybble
DC00-DC0F	56320-56335	Interfaz chip 1,IRQ
DD00-DD0F	56576-56591	Interfaz chip 2,NMI
D000-DFFF	53248-57294	Alternativa:set de caract.
E000-FFFF	57344-65535	ROM: sistema operativo
E000-FFFF	57344-65535	Alternativa: RAM
FF81-FFF5	65409-65525	Tabla Jump

MISCELANEAS

> TRABAJANDO CON CHR$

Cualquier codigo CHR$ o una combinación de codigos pueden ser asignados a un string:

```
100 C$=CHR$(147): REM LIMPIAR PANTALLA
200 PRINT C$"HOLA"
```

----*----

> EJECUCION DE COMANDOS DIRECTOS DESDE UN PROG.

Es facil hacer a un programa simular que determinados caracteres han sido tipeados. La idea es pokear mediante el programa los valores CHR$ de uno o mas caracteres en el area de memoria en el buffer del teclado (ver mapa de memoria en capitulo 7). Cuando el programa haya terminado su ejecucion la computadora escribira READY y luego respondera a los caracteres contenidos en el buffer como si hubieran sido tipeados en modo directo por Ud.
El buffer del teclado ocupa las direcciones desde 631 hasta 640. Trabaja en conjuncion con la direccion 198, la cual lleva la cuenta del numero de caracteres pokeados en buffer.
Corra el siguiente programa:

```
10 FOR I=1 TO 10:POKE 630+I,64+I
20 NEXT:POKE 198,10
```

Luego de READY apareceran las diez primeras letras del abecedario.
La aplicacion interesante de esto es que si Ud. pokea los caracteres de un comando ejecutable como RUN, LOAD, LIST, etc entonces estos podran ejecutarse automaticamente si agrega al buffer el

CHR$ correspondiente a la tecla RETURN que es el valor 13. Ver apendice de codigos ASCII y CHR$ del Manual del Usuario.

----*----

> LISTANDO DESDE UN PROGRAMA

Si Ud. ejecuta un comando LIST desde un programa el listado se efectuarà, pero la ejecución del programa se detendrá. Ud. puede utilizar la siguiente rutina para pokear el comando RUN en el buffer del teclado y ejecutar el RUN ni bien el listado se termine.

```
100 FOR I=631 TO 634:READ A:POKEI,A
200 NEXT:POKE198,4
300 DATA 82,85,78,13
400 LIST
```

----*----

> CARGA AUTOMATICA

Cuando Ud. desee cargar un programa desde otro en forma automatica puede utilizar:

```
10 PRINT"LOAD"CHR$(34)"nombre"CHR$(34)
   ",8[3 arriba]"
20 POKE631,13:POKE632,82:POKE633,85:
   POKE634,78:POKE635,13:POKE198,5:END
```

Cuando esto sea ejecutado, actuará como si Ud. hubiera tipeado el comando LOAD y luego tipeado RUN.

Si Ud. utiliza casstte reemplace el numero 8 por el 1.

----*----

> CARGANDO PROGRAMAS CON DATASSETTE
La manera mas fàcil de cargar y correr un programa desde otro es:

```
100 POKE631,131:POKE198,1
```

Esto produce idéntico efecto que presionado simultaneamente SHIFT-RUN/STOP.

-----*-----

> SUPRESION DEL "?" EN LOS INPUT
Se trata de utilizar el teclado como un "device". El numero de device correspondiente al teclado es cero (0) y puede tener un canal abierto para si.

```
10 OPEN1,0
20 PRINT "ESCRIBA ALGO";
30 INPUT#1,A$
40 PRINT
50 CLOSE 1
60 PRINT "UD. HA ESCRITO: "A$
```

La linea 10 abre un canal de entrada (INPUT); la 40 es necesaria para mover el cursor a la pròxima linea y la linea 50 cierra el canal.
Con este método se elimina el signo de interrogaciòn pero Ud. debe controlar cuidadosamente la posiciòn del cursor.

Otra manera de evitar el signo de interrogacion es utilizando GET en lugar de INPUT:

```
10 B$=""
20 POKE 207,0:POKE 204,0:GET A$:IF A$=""
   THEN 10
30 IF A$=CHR$(13) THEN PRINT CHR$(32):GOTO 40
```

```
40 PRINT A$;:B$=B$+A$:GOTO 20
50 REM  LO INGRESADO ES GUARDADO COMO B$
```

Es necesario PRINT CHR$(32) (espacio) para apagar el cursor en caso que este encendido cuando Ud. presiona la tecla RETURN.

-----*-----

> UTILIZACION DE TECLAS SELECCIONADAS
La siguiente rutina le permitirá detectar si una tecla fue presionada dentro de un grupo determinado de caracteres, el que Ud. necesite utilizar.

```
10 GET A$:J=1:FORI=1TO4:
20 IF A$=MID$("ABCD",I,1) THEN J=I+1
30 NEXT
40 ON J GOTO 10,aaa,bbb.ccc,ddd
```

La linea 40 permite direccionar según la tecla presionada.
En el ejemplo aaa,bbb,ccc,ddd, son los numeros de linea a los cuales yo quiero derivar el programa, segun A,B,C o D sean presionadas.
ABCD es el grupo de caracteres seleccionado. Este grupo puede ser tan largo como quiera e inclusive puede incluir los codigos de las teclas de funcion.



> UTILIZANDO EL COMANDO GET
Un buen programador escribiria esto:

```
10 PRINT "TOQUE UNA TECLA PARA CONTINUAR"
20 GET A$:IF A$="" THEN 20
```

Un mejor Programador borrarà el buffer del teclado Primero. De esa manera ignorarà las teclas Previamente Presionadas:

```
10 PRINT "TOQUE UNA TECLA PARA CONTINUAR"
15 GET A$:IF A$<>"" THEN 15
20 GET A$:IF A$="" THEN 20
```

Otra alternativa Para vaciar el buffer es:

```
15 POKE 198,0
```



> COMANDO GET CON CURSOR
Una desventaja del comando GET es que no Provee el cursor Para indicar al usuario la Pregunta. He aqui un metodo Para obtener el cursor:

```
10 POKE 204,0
20 GET A$:POKE 207,0:IF A$="" THEN 20
30 POKE 204,1:PRINT A$
```

El POKE en las lineas 10 y 20 le indican a la computadora que encienda el cursor mientras espera que una tecla sea Presionada. La linea 30 apaga el cursor e imprime el valor de A$. Serà necesario tocar Por lo menos la barra esPaciadora Para no dejar el cursor en Pantalla.



> SOLUCION DE ERRORES DE DATA
Si Ud. obtiene un ILLEGAL QUANTITY ERROR en una sentencia DATA cuando corre el Programa, tiPee lo

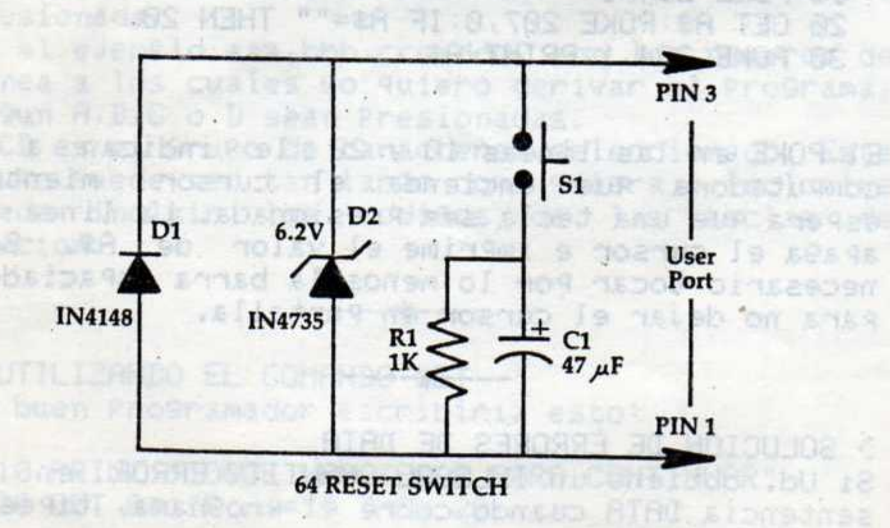
siguiente y obtendra el numero de linea donde se encuentra el error:

```
PRINT PEEK(63)+256*PEEK(64)
```

----*----

> RESET

Cuando a Ud. le suceda que un programa bloquee su computadora y las teclas RUN/STOP - RESTORE no devuelvan el cursor, sera el momento de usar un RESET (aplicable en casos de lenguaje de maquina).
Un reset basicamente es un switch que conecta los pins 1 y 3 del User Port (ubicado en la parte trasera derecha, de 12 pins a contar de izquierda a derecha, es de decir que el pin 12 es el que se encuntra mas al borde de la maquina). A continuacion se muestra el circuito en el cual se ha contemplado la eliminaciòn de todo riesgo para la computadora.



64 RESET SWITCH

> CONVERSION HEXA/DECIMAL

Trabajando con la C-64 el usuario desde sus primeros pasos en computación encontrarà a menudo nùmeros hexadecimales (base 16) muy utilizados, fundamentalmente en los programas en lenguaje de màquina, frecuentemente utilizados como sub-rutina en programas BASIC.

A continuaciòn mostramos una rutina que le permitirà realizar ambas conversiones, es decir de decimal a hexadecimal y viceversa.
Cuando convierta hexadecimal a decimal ingrese un numero de cuatro digitos, usando ceros cuando sea necesario. Por ejemplo, $9A sera 009A.

```
10 PRINT " 1- DEC A HEX"
20 PRINT " 2- HEX A DEC"
30 N$="0123456789ABCDEF"
40 PRINT:INPUT A
50 IF A=2 THEN 130
60 IF A<>1 THEN 10
70 PRINT"INGRESE # DEC";:INPUT A:B=1:C=3
80 D=16↑C:PRINT A;:PRINT "= $";:A=A+1
90 IF A-D>0 THEN A=A-D:B=B+1:GOTO 90
100 PRINT MID$(N$,B,1);:B=1:C=C-1:D=16↑C
110 IF C>-1 THEN 90
120 PRINT:PRINT:GOTO 10
130 PRINT "INGRESE # HEX";:INPUT H$:D=0:Q=3
140 FOR M=1 TO 4:FOR W=0 TO 15
150 IF MID$(H$,M,1)=MID$(N$,W+1,1) THEN 170
160 NEXT W
170 D1=W*(16↑(Q)):D=D+D1:Q=Q-1:NEXT M
180 DE=INT(D):PRINT"$";H$;" =";DE
190 PRINT:GOTO 10
```

INDICE